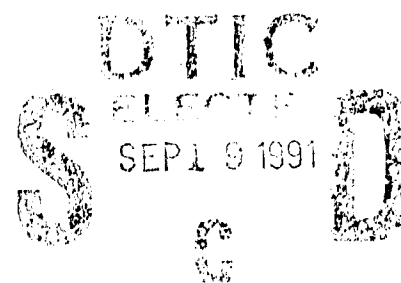


AB-A246 667

SSD-TR-91-27



AEROSPACE REPORT NO.
TR-0091(6925-05)-3



A Conjugate Gradient Based Algorithm to Minimize the Sidelobe Level of Planar Arrays with Element Failures

Prepared by

T. J. PETERS
Communications Systems Subdivision

31 August 1991

Prepared for

SPACE SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
Los Angeles Air Force Base
P. O. Box 92960
Los Angeles, CA 90009-2960

Engineering and Technology Group

91-11030



THE AEROSPACE CORPORATION
El Segundo, California

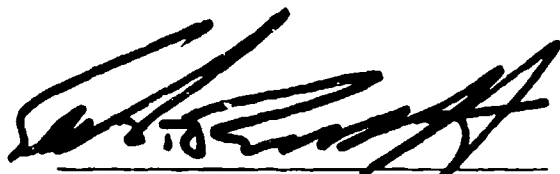
91 0 18 085

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED

This report was submitted by The Aerospace Corporation, El Segundo, CA 90245-4691, under Contract No. F04701-88-C-0089 with the Space Systems Division, P. O. Box 92960, Los Angeles, CA 90009-2960. It was reviewed and approved for The Aerospace Corporation by J. M. Straus, Principal Director, Communications Systems Subdivision.

Seth Parkoff, Capt, USAF, was the project officer for the Mission-Oriented Investigation and Experimentation (MOIE) program. This report has been reviewed by the Public Affairs Office (PAS) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication. Publication of this report does not constitute Air Force approval of the report's findings or conclusions. It is published only for the exchange and stimulation of ideas.



SETH PARKOFF, Capt, USAF
MOIE Project Officer
SSD/MHE



JONATHAN M. EMMES, Maj, USAF
MOIE Program Manager
PL/WCO OL-AH

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION Unclassified			1b RESTRICTIVE MARKINGS		
2a SECURITY CLASSIFICATION AUTHORITY			3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution Unlimited		
2b DECLASSIFICATION/DOWNGRADING SCHEDULE					
4 PERFORMING ORGANIZATION REPORT NUMBER(S) TR-0091(6925-05)-3			5 MONITORING ORGANIZATION REPORT NUMBER(S) SSD-TR-91-27		
6a NAME OF PERFORMING ORGANIZATION The Aerospace Corporation Communications Systems Subdivision		6b OFFICE SYMBOL (If applicable)	7a NAME OF MONITORING ORGANIZATION Space Systems Division		
6c ADDRESS (City, State, and ZIP Code) P. O. Box 92957 Los Angeles, CA 90009-2957			7b ADDRESS (City, State, and ZIP Code) Los Angeles Air Force Base P. O. Box 92960 Los Angeles, CA 90009-2960		
8a NAME OF FUNDING/SPONSORING ORGANIZATION		8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F04701-88-C-0089		
8c ADDRESS (City, State, and ZIP Code)			10 SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.
			WORK UNIT ACCESSION NO.		
11 TITLE (Include Security Classification) A Conjugate Gradient Based Algorithm to Minimize the Sidelobe Level of Planar Arrays with Element Failures					
12 PERSONAL AUTHOR(S) Peters, T. J.					
13a TYPE OF REPORT		13b TIME COVERED FROM _____ TO _____		14 DATE OF REPORT (Year, Month, Day) 31 August 1991	
15 PAGE COUNT 83					
16 SUPPLEMENTARY NOTATION					
17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP	Antenna array synthesis		
			Conjugate gradient method		
19 ABSTRACT (Continue on reverse if necessary and identify by block number) Element failures increase the sidelobe power level of an array. Reconfiguring the amplitude and phase of the remaining elements can partially compensate for the failed elements and thus reduce the sidelobe level. The result of this investigation is an algorithm that yields the reconfigured distribution by minimizing the ratio of the average peak sidelobe power level to the power in the mainbeam, taking into account the defective elements. The minimization of this nonlinear function is carried out via a conjugate gradient method. The algorithm is applied to the synthesis of sum and difference patterns of planar arrays.					
20 DISTRIBUTION AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED UNLIMITED <input type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS			21 ABSTRACT SECURITY CLASSIFICATION Unclassified		
22a NAME OF RESPONSIBLE INDIVIDUAL			22b TELEPHONE (Include Area Code)		22c OFFICE SYMBOL

Preface

The author would like to thank Dr. Keith M. Soo Hoo, Dr. Robert B. Dybdal and Don J. Hinshilwood for their helpful comments.



Approved for	
DATE	06/01/64
BY	Tab
Signature	
Justification	
by	
Classification	
• Unclassified	
Excluded from	
Control	

A-1

Contents

Preface	1
I. Introduction	7
II. Power Pattern With Element Failures	9
III. Minimizing Nonlinear Functions	13
IV. Sum Pattern Synthesis	19
V. Difference Pattern Synthesis	25
VI. Results and Discussion	27
VII. Conclusion	37
References	39
Appendix: Fortran 77 Programs	41

Figures

1. Independently Controlled Elements for (a) Sum Pattern and (b) Difference Pattern	10
2. One-Dimensional Minimization	14
3. One-Dimensional Average Sidelobe Accumulation	21
4. Sum Pattern Sidelobe Sample Region	22
5. Difference Pattern Sidelobe Sample Region	26
6. Hexagonal 91 Element Array with 3 Element Failures	28
7. Sum Power Pattern with No Failures	29
8. Sum Power Pattern with 3 Element Failures	30
9. Synthesized Sum Power Pattern with 3 Element Failures	31
10. Difference Power Pattern with No Failures	32
11. Difference Power Pattern with 3 Element Failures	33
12. Synthesized Difference Power Pattern with 3 Element Failures	34

I. Introduction

An antenna array is composed of many elements whose excitation amplitude and phase can be individually adjusted to yield a desired pattern. If all array elements operate properly, then well-known analytic techniques can be used to find the optimum amplitude and phase of each element, to yield a given beamwidth and sidelobe level. However, if any elements fail, then no analytic means exists to find the aperture distribution that compensates for the degradation of the pattern. Element failures destroy symmetry and cause sharp variations in the field intensity across the array aperture, increasing the sidelobe level of the power pattern. The simplest solution to this problem is to increase the taper of the array distribution in order to lower the sidelobes back to the design level. Unfortunately, this solution has several disadvantages. First, as the design sidelobe level for an array decreases, the sensitivity of the sidelobe level to a perturbation in an excitation value increases. Second, changing the design taper may broaden the main beam to an unacceptable level. Finally, merely increasing the taper does not compensate for lack of symmetry or smooth out sharp field variations. Compensation for the defective elements can be achieved, by numerically finding the excitation of each nondefective element that optimizes some function. This function must take into account the location of the failed elements.

The approach used in this study is to reformulate the optimization problem to take into account the element failures. An algorithm is developed that synthesizes the amplitude and phase of each nondefective element in order to produce the lowest sidelobe level for a given beamwidth. This algorithm can also be used to find the element failure limits of any array. More specifically, the algorithm can find the maximum number of elements beyond which it is impossible to recover lost performance. Also, the loss in performance can be determined by the geometric arrangement of the failed elements. The numerical implementation of the synthesis algorithm involves the minimiza-

tion of a nonlinear function, which is the ratio of the sum of a set of peak sidelobe power points to the power in the main beam. This minimization will be carried out via a conjugate gradient method.

II. Power Pattern With Element Failures

This analysis neglects the element factor and considers only the array factor. No mutual coupling effects are modeled. Failed elements are deleted from the array factor. The $e^{j\omega t}$ time convention is understood, where ω is the angular frequency of the excitation with a freespace wavelength of λ_0 . Letting N denote the number of nonfailed elements, the array factor (AF) is given by

$$AF = \sum_{n=1}^N I_n e^{j(k_x x_n + k_y y_n)} \quad (1)$$

where

$$k_x = k_0 \sin(\theta) \cos(\phi) \quad (2)$$

$$k_y = k_0 \sin(\theta) \sin(\phi) \quad (3)$$

and $k_0 = 2\pi/\lambda_0$. The excitation of the n th element, denoted by I_n , can be expressed by the complex quantity $I_n = u_n + jv_n$, where $\sqrt{u_n^2 + v_n^2}$ yields the amplitude and $\tan^{-1}(v_n/u_n)$ yields the phase angle. The power pattern is proportional to the square of the magnitude of AF.

The array is assumed to operate in either sum or difference mode. The sum mode produces a peak in the pattern in the broadside direction and the difference mode produces a null in the broadside direction. A possible element configuration to generate both types of patterns is shown in Fig. 1. Sum mode operation allows all 37 elements to be independently controlled, as shown in Fig. 1(a). The same array operating in difference mode has only 17 independently controlled elements, as shown in Fig. 1(b). This is because half of the array elements are excited with their phase opposite from that of the corresponding elements symmetric with respect to the center of the array. If an element fails, then the symmetric counterpart is deactivated. This ensures a stationary beam null and a symmetric beam slope, independent of the synthesis technique used.

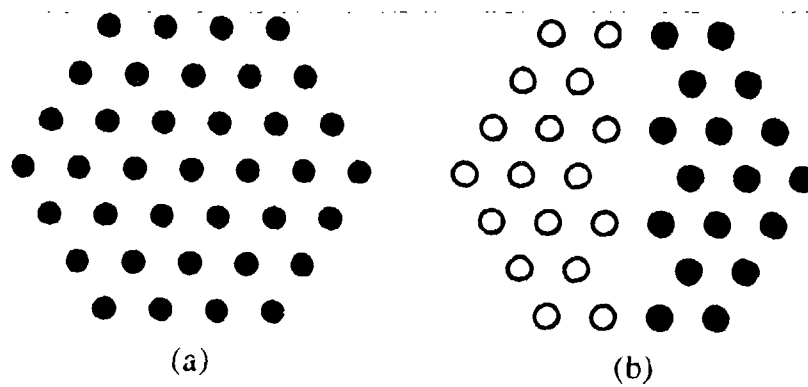


Fig. 1. Independently Controlled Elements for (a) Sum Pattern and (b) Difference Pattern

The synthesis of the excitation values has three possible combinations: synthesizing with amplitude only, synthesizing with phase only, and synthesizing with both amplitude and phase. It is well known that synthesizing the lowest possible sidelobe level for a given beamwidth over a symmetric aperture yields a uniform phase and a nonuniform amplitude distribution. However, if the failed elements yield a nonsymmetric array distribution, then the optimal phase may not be constant. Phase synthesis with a constant amplitude was found by Deford and Gandhi (Ref. 1) to require an extremely large number of elements to yield low sidelobes. A similar analysis by Peters (Ref. 2) found that phase-only synthesis was not very effective in reducing sidelobes for fixed-amplitude arrays. Therefore, both amplitude and phase synthesis would seem to be required for the generation of a sum pattern. Since the difference pattern preserves the symmetry, amplitude-only synthesis is required. The sum power pattern is given by

$$P(\theta, \phi, u, v) = \left[\sum_{n=1}^N [u_n \cos(\psi_n) - v_n \sin(\psi_n)] \right]^2 + \left[\sum_{n=1}^N [u_n \sin(\psi_n) + v_n \cos(\psi_n)] \right]^2 \quad (4)$$

where $v_n = k_x x_n + k_y y_n$ and the difference power pattern may be written as

$$P(\theta, \phi, u) = \left[\sum_{n=1}^N u_n \sin(\psi_n) \right]^2 \quad (5)$$

Note that the amplitude is allowed to be negative on the difference pattern, since that involves only a simple phase shift.

III. Minimizing Nonlinear Functions

Equations (4) and (5) indicate that the power pattern of the array is a nonlinear function of the amplitude and phase of the excitation. Minimizing the peak sidelobe level will require a mathematical technique that can handle a nonlinear function. A multivariable quadratic function can be minimized by the conjugate gradient algorithm developed by Hestenes and Stiefel (Ref. 3). An arbitrary nonlinear function can also be minimized by the same algorithm if a suitable quadratic approximation can be found. Although several variations are given in the literature, all have the same two major steps discussed by Hestenes (Ref. 4). First, the function must be approximated by a quadratic truncation of a Taylor series expanded around an initial estimate of the solution. Second, this quadratic is minimized by the conjugate gradient algorithm to obtain a new estimate. This new estimate is in turn used as the expansion point for a new quadratic approximation. Since the quadratic approximation is valid only near the point of expansion, finding a minimum of this quadratic is the same as finding a minimum of the original function only if both have the same minimum. Therefore, the algorithm must be restarted so that the quadratic approximation can ultimately be expanded around a point that approaches the minimum of the original function. The one-dimensional case is illustrated in Fig. 2. Given the function $f(s)$ and the initial guess s_0 , the quadratic $q_0(s)$ is formed and given by

$$q_0(s) = f(s_0) + f'(s_0)(s - s_0) + \frac{1}{2}f''(s_0)(s - s_0)^2. \quad (6)$$

The minimum of this quadratic is denoted by s_1 and is given by the Newton estimate

$$s_1 = s_0 - \frac{f'(s_0)}{f''(s_0)}. \quad (7)$$

The estimate s_1 is then used to compute $q_1(s)$. The iteration then continues until the minimum of the quadratic approximation coincides with the minimum of the original function.

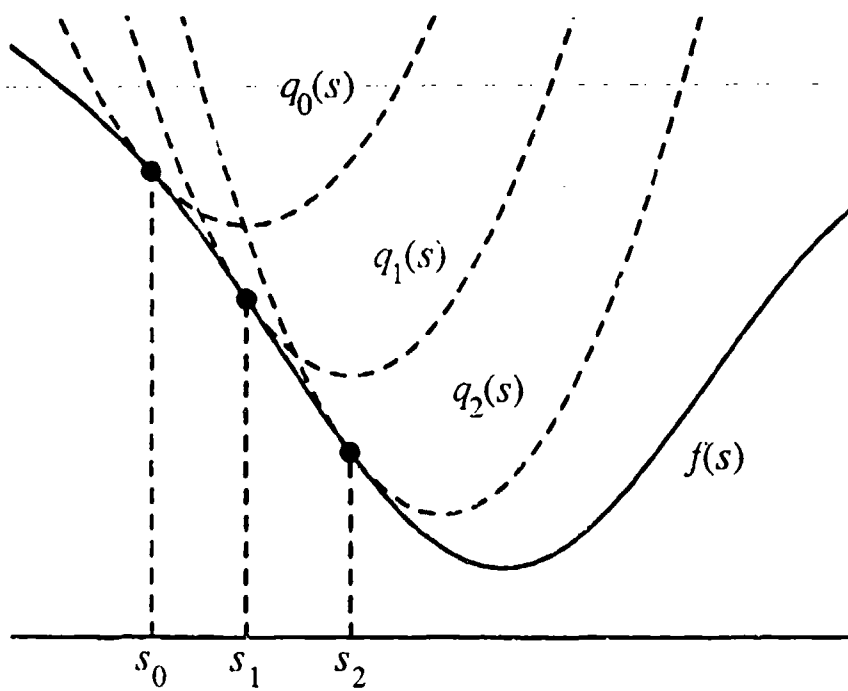


Fig. 2. One-Dimensional Minimization

The principles of the one-dimensional minimization apply also to multidimensional functions. For example, consider the function $f(u, v)$, where u and v are $N \times 1$ column vector variables. The quadratic approximation may be written in matrix form as

$$f(u, v) = f(u^1, v^1) + [\gamma - \gamma^1]^T G + \frac{1}{2} [\gamma - \gamma^1]^T H [\gamma - \gamma^1] \quad (8)$$

where T denotes the transpose and the components are defined as follows. The vector γ is a $2N \times 1$ column vector defined by $\gamma = [u \ v]^T$. The gradient, denoted by G , is a $2N \times 1$ column vector evaluated at (u^1, v^1) and is given by $G = [G_u \ G_v]^T$, where each subvector has the form

$$G_u = \left[\frac{\partial f}{\partial u_1} \quad \frac{\partial f}{\partial u_2} \quad \dots \quad \frac{\partial f}{\partial u_N} \right]. \quad (9)$$

The Hessian, denoted by H , is a $2N \times 2N$ matrix whose elements are evaluated at (u^1, v^1) and expressed as

$$H = \begin{bmatrix} H_{uu} & H_{uv} \\ H_{vu} & H_{vv} \end{bmatrix} \quad (10)$$

where each submatrix has the form

$$H_{uv} = H_{vu}^T = \begin{bmatrix} \frac{\partial^2 f}{\partial u_1 \partial v_1} & \frac{\partial^2 f}{\partial u_1 \partial v_2} & \dots & \frac{\partial^2 f}{\partial u_1 \partial v_N} \\ \frac{\partial^2 f}{\partial u_2 \partial v_1} & \frac{\partial^2 f}{\partial u_2 \partial v_2} & \dots & \frac{\partial^2 f}{\partial u_2 \partial v_N} \\ \vdots & \vdots & \dots & \vdots \\ \frac{\partial^2 f}{\partial u_N \partial v_1} & \frac{\partial^2 f}{\partial u_N \partial v_2} & \dots & \frac{\partial^2 f}{\partial u_N \partial v_N} \end{bmatrix}. \quad (11)$$

Using the one-dimensional analogy, the minimum of Eqn. (8) may be computed from the matrix form of Newton's formula, as given by

$$\gamma_1 = \gamma_0 + H^{-1}(\gamma_0)G(\gamma_0). \quad (12)$$

However, this requires that the inverse of H be computed. In many cases of practical interest, only an estimate of the minimum of the quadratic is required; therefore, it is more efficient to use the conjugate gradient method to obtain an estimate of this minimum. Making the substitution $z_u = u - u^1$,

$z_v = v - v^1$, and $D = -G$ into Eqn. (8) and excluding the constant yields a standard quadratic function, denoted by c and written as

$$c = \frac{1}{2} [z_u \quad z_v] \begin{bmatrix} H_{uu} & H_{uv} \\ H_{vu} & H_{vv} \end{bmatrix} \begin{bmatrix} z_u \\ z_v \end{bmatrix} - [z_u \quad z_v] \begin{bmatrix} D_u \\ D_v \end{bmatrix}. \quad (13)$$

Minimizing this expression is the same as solving the linear system

$$\begin{bmatrix} H_{uu} & H_{uv} \\ H_{vu} & H_{vv} \end{bmatrix} \begin{bmatrix} z_u \\ z_v \end{bmatrix} = \begin{bmatrix} D_u \\ D_v \end{bmatrix} \quad (14)$$

where the Hessian matrix H is always symmetric and nonsingular but not always positive definite. Therefore, the conjugate gradient algorithm must be general enough to handle this situation. A suitable algorithm with L restarts and $K \ll N$ conjugate gradient iterations per restart is given as follows (Ref. 5):

begin loop for $l=1, \dots, L$

compute D^l , H^l and set $z_u^1 = 0, z_v^1 = 0$

$$r_{u,v}^1 = D_{u,v}^l \quad (15)$$

$$q_u = H_{uu} r_u^1 + H_{uv} r_v^1 \quad (16)$$

$$q_v = H_{vu} r_u^1 + H_{vv} r_v^1 \quad (17)$$

$$\beta_0 = \frac{1}{\langle q_u, q_u \rangle + \langle q_v, q_v \rangle} \quad (18)$$

$$p_{u,v}^1 = \beta_0 q_{u,v} \quad (19)$$

begin loop for $k=1, \dots, K$

$$q_u = H_{uu} p_u^k + H_{uv} p_v^k \quad (20)$$

$$q_v = H_{vu} p_u^k + H_{vv} p_v^k \quad (21)$$

$$\alpha_k = \frac{1}{\langle q_u, q_u \rangle + \langle q_v, q_v \rangle} \quad (22)$$

$$z_{u,v}^{k+1} = z_{u,v}^k + \alpha_k p_{u,v}^k \quad (23)$$

$$r_{u,v}^{k+1} = r_{u,v}^k - \alpha_k q_{u,v} \quad (24)$$

$$crr = \frac{\langle r_{u,v}^{k+1}, r_{u,v}^{k+1} \rangle + \langle r_{u,v}^k, r_{u,v}^k \rangle}{\langle r_{u,v}^k, r_{u,v}^k \rangle + \langle r_{u,v}^{k+1}, r_{u,v}^{k+1} \rangle} \quad (25)$$

if $\text{err} < \text{tolerance}$, terminate k loop, else

$$q_u = H_{uu}r_u^{k+1} + H_{uv}r_v^{k+1} \quad (26)$$

$$q_v = H_{vu}r_u^{k+1} + H_{vv}r_v^{k+1} \quad (27)$$

$$\beta_k = \frac{1}{\langle q_u, q_u \rangle + \langle q_v, q_v \rangle} \quad (28)$$

$$p_{u,v}^{k+1} = p_{u,v}^k + \beta_k q_{u,v} \quad (29)$$

—continue k loop

$$u, v^{l+1} = u, v^l + z_{u,v}^{k+1} \quad (30)$$

continue l loop

The l loop determines the quadratic approximation to the function $f(u, v)$. The gradient and Hessian are computed at the current estimate and the appropriate conjugate gradient parameters are initialized. The k loop obtains an estimate of the minimum of c by computing an estimate of the solution to the linear system described by Eqn. (14). The dominant computational effort for each k iteration is caused by the matrix vector products, which require $\mathcal{O}(8N^2)$ multiplications. The computation of the Hessian requires $\mathcal{O}(\frac{1}{2}TN^2)$ multiplications, where T is the number of multiplications per matrix element. Since T depends on the function, the computational efficiency of the overall algorithm is function-dependent. An alternate algorithm used by Fong and Birgenheier (Ref. 6) and Press *et al.* (Ref. 7) does not require the computation of the Hessian, but implicitly assumes that it is positive definite. This can lead to an unpredictable breakdown in the algorithm.

IV. Sum Pattern Synthesis

The goal of the sum pattern synthesis is to maximize the power in the main beam direction while simultaneously minimizing the power in the peak sidelobe. Unfortunately, forcing the sidelobe to a lower level at a particular angular location without any constraint on the other sidelobes will merely shift the peak sidelobe to a new location. Because the peak sidelobe location may change, there is no mathematically elegant way to achieve the goal directly. Rather, the same objective may be achieved indirectly, by minimizing the average of the accumulated peak sidelobe power points. The average is defined to be the average of these peak sidelobes, computed at each restart of the minimization algorithm. If M_l denotes the number of distinct peak sidelobe points at the l th restart, then the average power, denoted by $P_a(u, v)$, is defined by

$$P_a(u, v) = \frac{1}{M_l} \sum_{i=1}^{M_l} P_i. \quad (31)$$

The power at the i th angular point is defined as $P_i = P(u, v, \theta_i, \phi_i)$, where $i = 0$ denotes the main beam location and $i = 1 \dots M$ denotes peak sidelobe points. In order to minimize the sidelobe power while simultaneously maximizing the main beam power, a function $f(u, v)$ is defined as the ratio of the average peak power to the main beam power and is expressed as

$$f(u, v) = \frac{P_a(u, v)}{P_0(u, v)}. \quad (32)$$

Minimizing this function will achieve the desired goal. A possible one-dimensional scenario is illustrated in Figure 3. The peak sidelobe level is found at the initial start. This sidelobe level is then reduced by a partial minimization of $f(u, v)$. At the first restart the peak sidelobe point is still at the same location, so that peak is reduced again. At the second restart a new peak sidelobe location is found, so this peak and the previous peak are averaged and reduced. At the third restart the peak location has changed

again, so the average is updated and $f(u, v)$ is minimized. At the fourth restart another point is added to the average and $f(u, v)$ is minimized. The pattern has reached the steady-state power distribution.

An inherent characteristic of this method is that all the sidelobes will tend to be brought to the same level. It should be noted that all the sidelobe sample points could have been used a priori at each restart without searching for the peak sidelobe point and the results would be the same. However, using the accumulated average has several advantages. The function $f(u, v)$ changes rapidly when few sidelobe points are present, so that the peak may be brought down fast. As more peak points are added to the average, a self-damping effect takes place and reduces the sensitivity of the pattern to changes in u and v , allowing a smooth transition to an optimum. Accumulating the peak points is numerically much faster than starting with all points, since the computation time for evaluating $f(u, v)$ is dependent on M_l .

The sidelobe region is defined as the hemisphere formed by the limits $0 \leq \phi \leq 2\pi$ and $0 \leq \theta \leq \pi$ minus the circular aperture of the main beam up to the first null. The density of the sample points should be chosen in such a way that the peak sidelobe may be accurately estimated. A good rule of thumb is that the sample separation distance should be no greater than half the main beamwidth of the design pattern. A uniformly spaced square lattice is sufficient and is shown in Fig. 4. The parameter k_m is the radius of the main beam null of the design pattern, increased slightly to allow the beam to broaden as the sidelobe level is reduced.

The components of D and H required for the conjugate gradient method are given as follows:

$$D_u^m = -\frac{\partial f}{\partial u_m} = \frac{1}{P_0} \left[\frac{P_a}{P_0} \frac{\partial P_0}{\partial u_m} - \frac{\partial P_a}{\partial u_m} \right] \quad (33)$$

$$D_v^m = -\frac{\partial f}{\partial v_m} = \frac{1}{P_0} \left[\frac{P_a}{P_0} \frac{\partial P_0}{\partial v_m} - \frac{\partial P_a}{\partial v_m} \right] \quad (34)$$

$$H_{u_n}^{m_n} = \frac{\partial^2 f}{\partial u_m \partial u_n} = \frac{1}{P_0} \left[D_u^m \frac{\partial P_0}{\partial u_n} + D_v^m \frac{\partial P_0}{\partial v_n} + \frac{\partial^2 P_a}{\partial u_m \partial u_n} - \frac{P_a}{P_0} \frac{\partial^2 P_0}{\partial u_m \partial u_n} \right] \quad (35)$$

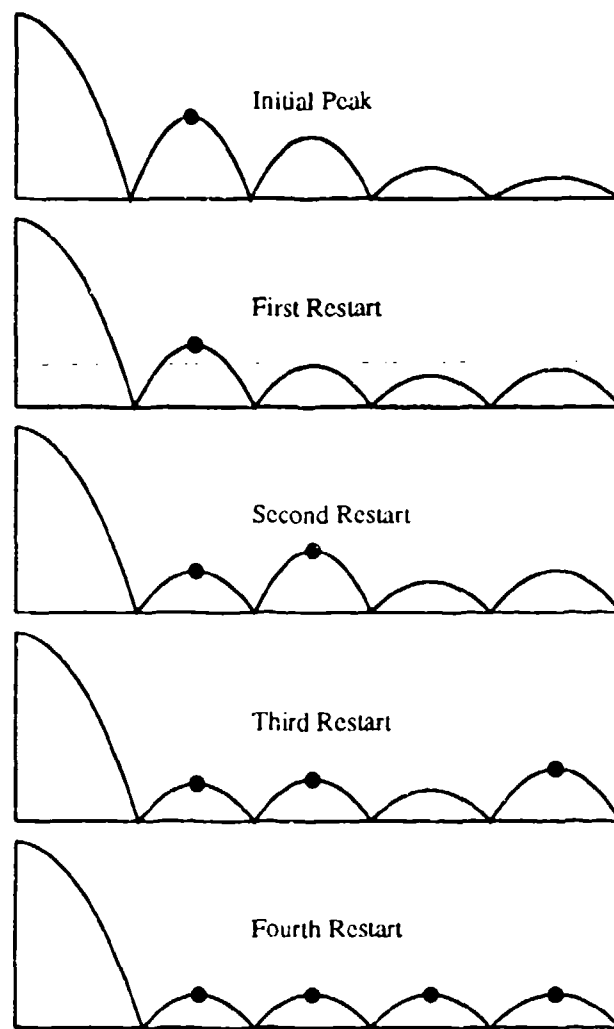


Fig. 3. One-Dimensional Average Sidelobe Accumulation

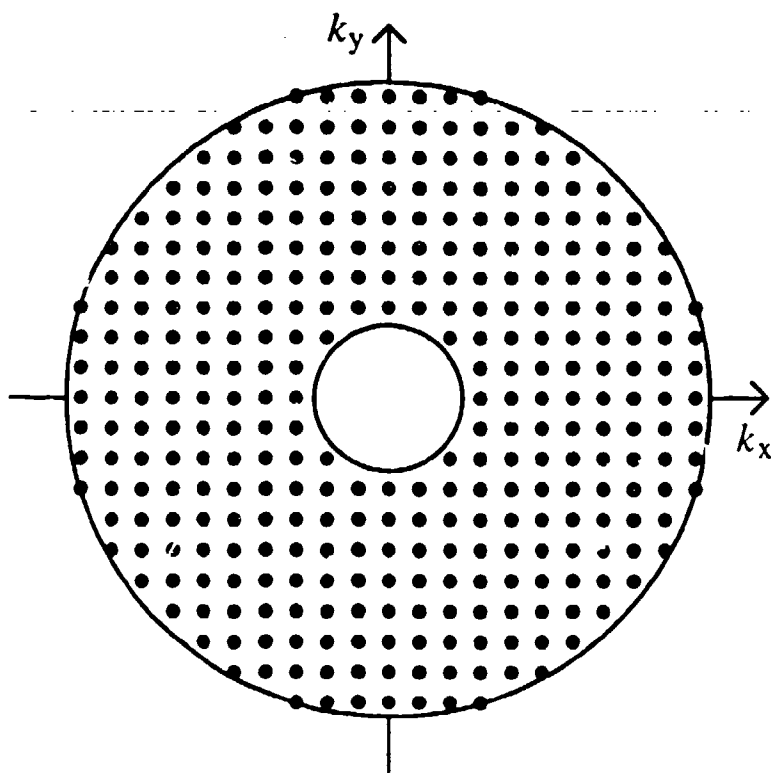


Fig. 4. Sum Pattern Sidelobe Sample Region

$$H_{uv}^{mn} = \frac{\partial^2 f}{\partial u_m \partial v_n} = \frac{1}{P_0} \left[D_v^n \frac{\partial P_0}{\partial u_m} + D_u^m \frac{\partial P_0}{\partial v_n} + \frac{\partial^2 P_a}{\partial u_m \partial v_n} - \frac{P_a}{P_0} \frac{\partial^2 P_0}{\partial u_m \partial v_n} \right] \quad (36)$$

$$H_{vu}^{mn} = H_{uv}^{mn} \quad (37)$$

$$H_{vv}^{mn} = \frac{\partial^2 f}{\partial v_m \partial v_n} = \frac{1}{P_0} \left[D_v^n \frac{\partial P_0}{\partial v_m} + D_v^m \frac{\partial P_0}{\partial v_n} + \frac{\partial^2 P_a}{\partial v_m \partial v_n} - \frac{P_a}{P_0} \frac{\partial^2 P_0}{\partial v_m \partial v_n} \right] \quad (38)$$

The power at the i th angle can be computed by introducing the auxiliary functions ξ_i and χ_i and defining $P_i = \xi_i^2 + \chi_i^2$ such that

$$\xi_i = \sum_{w=1}^N [u_w \cos(\psi_{wi}) - v_w \sin(\psi_{wi})] \quad (39)$$

$$\chi_i = \sum_{w=1}^N [u_w \sin(\psi_{wi}) + v_w \cos(\psi_{wi})] \quad (40)$$

where $\psi_{wi} = k_{xi}x_w + k_{yi}y_w$. The first-derivatives of the power used in the gradient are given by

$$\frac{\partial P_i}{\partial u_m} = 2 [\xi_i \cos(\psi_{mi}) + \chi_i \sin(\psi_{mi})] \quad (41)$$

$$\frac{\partial P_i}{\partial v_m} = 2 [\chi_i \cos(\psi_{mi}) - \xi_i \sin(\psi_{mi})] \quad (42)$$

and the second derivative terms used in the Hessian are expressed as

$$\frac{\partial^2 P_i}{\partial u_m \partial u_n} = 2 \cos(\psi_{mi} - \psi_{ni}) \quad (43)$$

$$\frac{\partial^2 P_i}{\partial u_m \partial v_n} = 2 \sin(\psi_{mi} - \psi_{ni}) \quad (44)$$

$$\frac{\partial^2 P_i}{\partial v_m \partial v_n} = \frac{\partial^2 P_i}{\partial u_m \partial u_n} \quad (45)$$

Note that the computation of the Hessian requires the evaluation of the power at the main beam location and at each peak sidelobe point. Since the power computation requires $\mathcal{O}(N)$ multiplications, the multiplications per matrix element is $T = \mathcal{O}(M_i N)$. Therefore, the Hessian computation dominates the computation of a single l iteration.

V. Difference Pattern Synthesis

The synthesis procedure for the difference pattern is similar to that used for the sum pattern, except that now only amplitude synthesis is necessary. The appropriate function defined in Eqn. (32) is reduced to one dimension as

$$f(u) = \frac{P_a(u)}{P_0(u)}. \quad (46)$$

The power at the i th location is defined as $P_i = \xi_i^2$, where

$$\xi_i = \sum_{v=1}^N u_v \sin(\psi_{vi}). \quad (47)$$

The first derivative of the power is computed from

$$\frac{\partial P_i}{\partial u_m} = 2\xi_i \sin(\psi_{mi}) \quad (48)$$

with the second derivative given by

$$\frac{\partial^2 P_i}{\partial u_m \partial u_n} = 2 \sin(\psi_{mi}) \sin(\psi_{ni}). \quad (49)$$

For the configuration shown in Fig. 1, the power pattern is symmetric across the $y - z$ plane. Thus, the k -space sample region need only contain points where $k_x > 0$, as shown in Fig. 5.

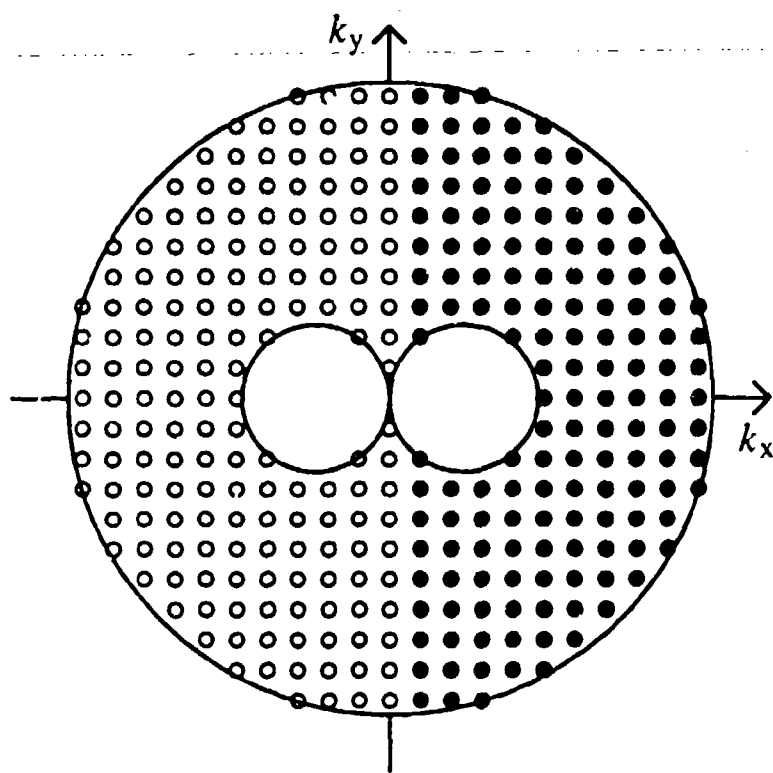


Fig. 5. Difference Pattern Sidelobe Sample Region

VI. Results and Discussion

A possible element failure scenario is shown in Fig. 6. The hexagonal array has a total of 91 elements spaced $0.6\lambda_0$ apart with 3 element failures. The desired aperture function for the sum pattern is a parabolic with power $N=2$ and an edge taper of -20 dB. This yields a -35 dB sidelobe level. The aperture function for the difference pattern is a Bayliss distribution with $n=10$ and a -40 dB sidelobe level. All the power patterns are plotted in dB using cylindrical coordinates, with θ as the radial coordinate in the range $0 \leq \theta \leq 90^\circ$. A reference -32 dB floor is placed on each plot.

The sum power pattern with no element failures is shown in Fig. 7. The peak sidelobe level is -30.9 dB with a peak gain of 20.8 dB. The main beam has azimuthal symmetry with a half-power beamwidth of 12.86° . The synthesized sum pattern of Fig. 8 required 300 restarts, with 5 conjugate gradient iterations per restart, to solve for the 176 required unknowns. The effect of the failures is to increase the sidelobe level to -27.4 dB and reduce the gain to 20.5 dB. The main beam becomes slightly elliptical with a minimum half-power beamwidth of 12.57° and a maximum of 13.00° . The synthesized pattern, shown in Fig. 9, has a peak sidelobe level of -31.3 dB and a gain of 19.5 dB. The main beam also has an elliptical cross section with a minimum half-power beamwidth of 13.71° and a maximum of 16.29° . As expected, the reduction in the sidelobe level produces a corresponding increase in the beamwidth and a drop in gain.

The difference power pattern with no element failures is shown in Fig. 10. The peak sidelobe level is -33.0 dB with a peak gain of 18.4 dB. The angular half-power null width between the peaks is 8.57° . Fig. 11 shows that the effect of the element failures is to increase the sidelobe level to -21.7 dB and drop the gain to 18.0 dB, with the half-power null width remaining about the same. The synthesized difference pattern of Fig. 12 required 100 restarts with 5 conjugate gradient iterations per restart to solve for the 80 unknowns.

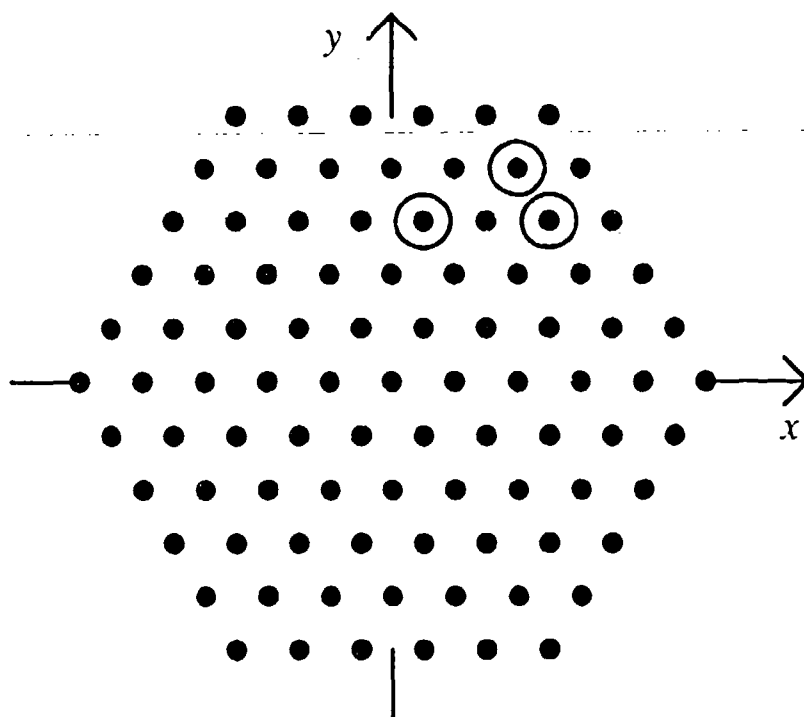


Fig. 6. Hexagonal 91 Element Array with 3 Element Failures

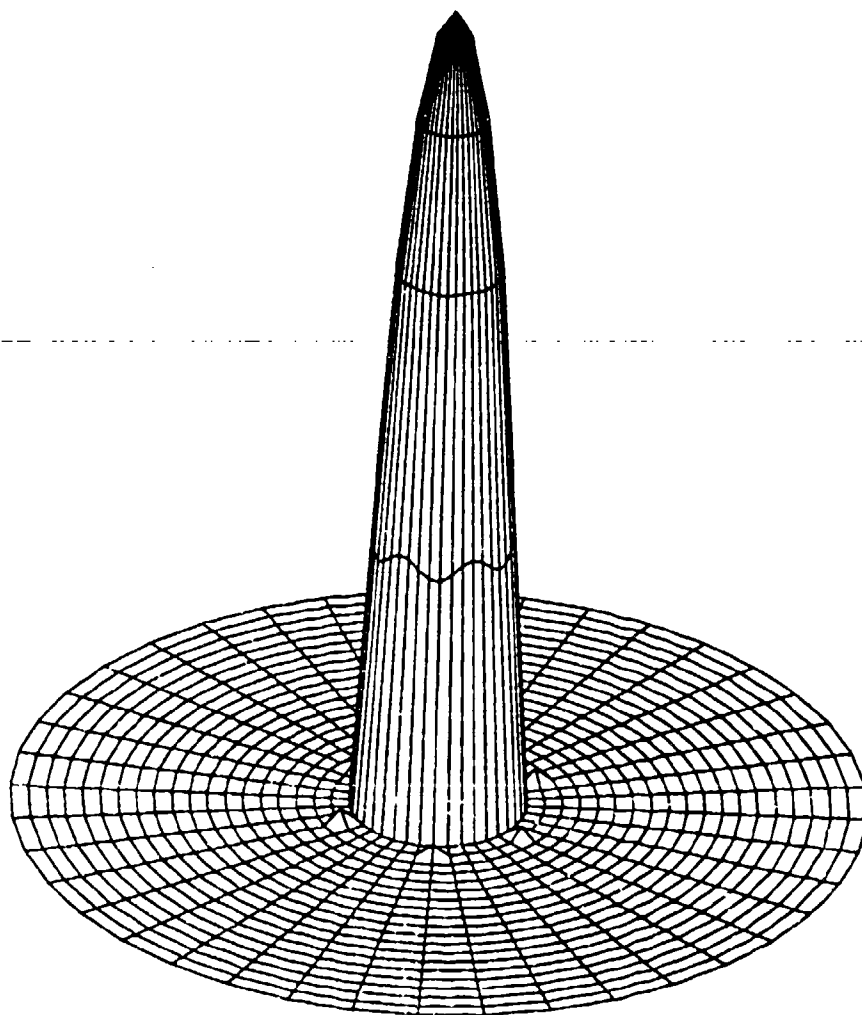


Fig. 7. Sum Power Pattern with No Failures

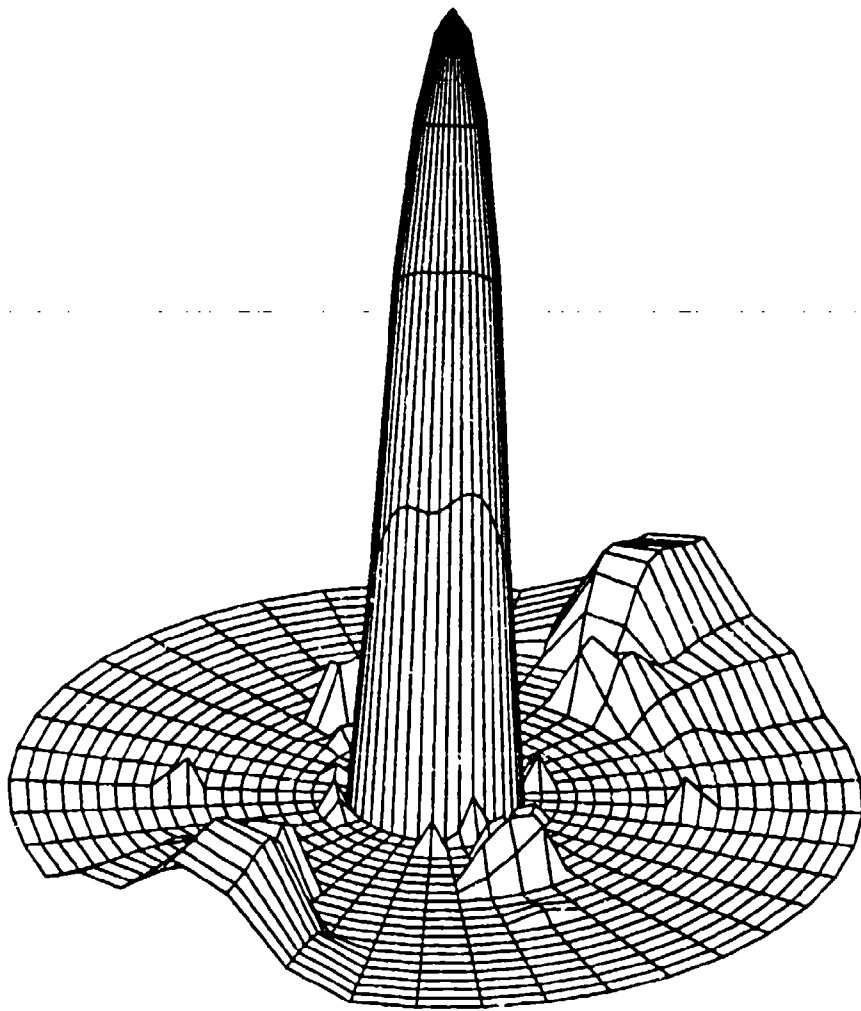


Fig. 8. Sum Power Pattern with 3 Element Failures

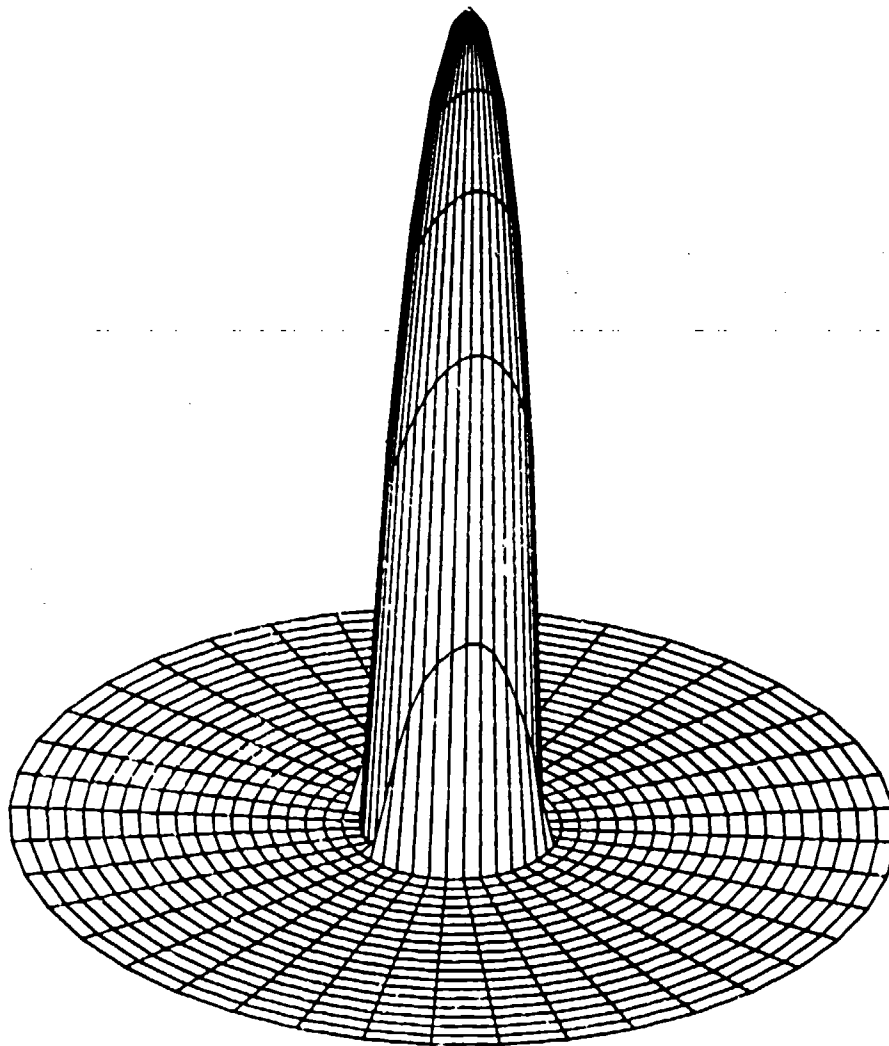


Fig. 9. Synthesized Sum Power Pattern with 3 Element Failures

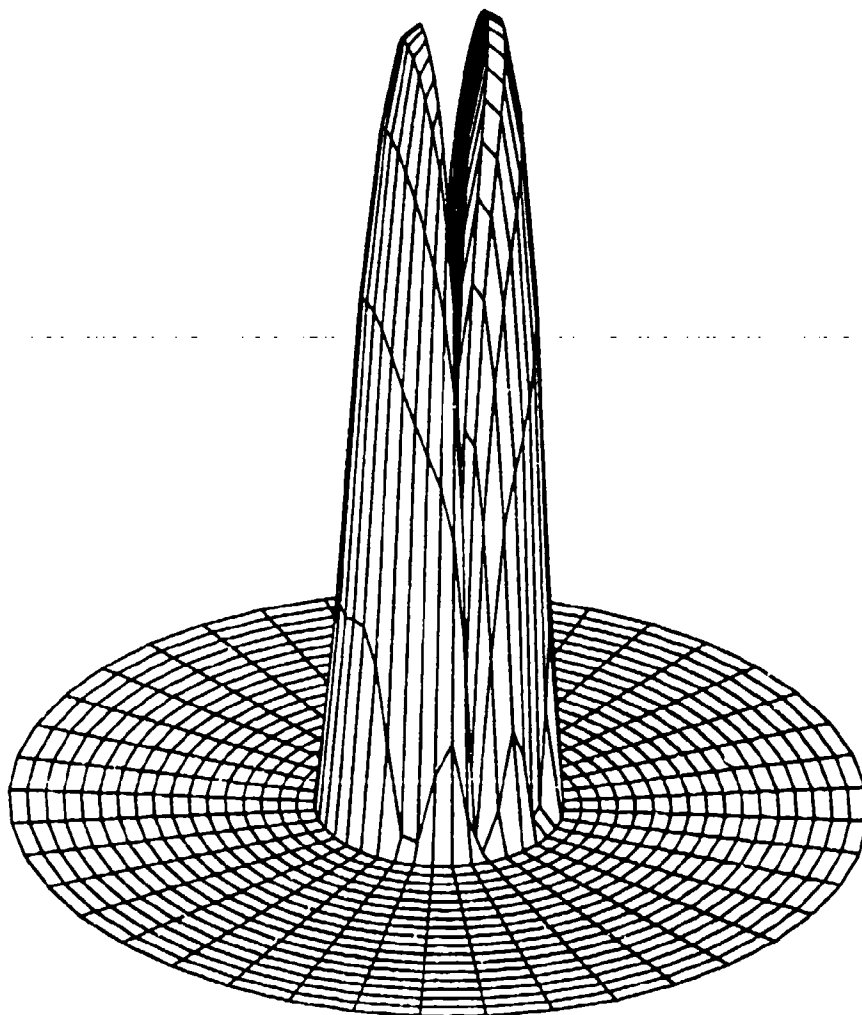


Fig. 10. Difference Power Pattern with No Failures

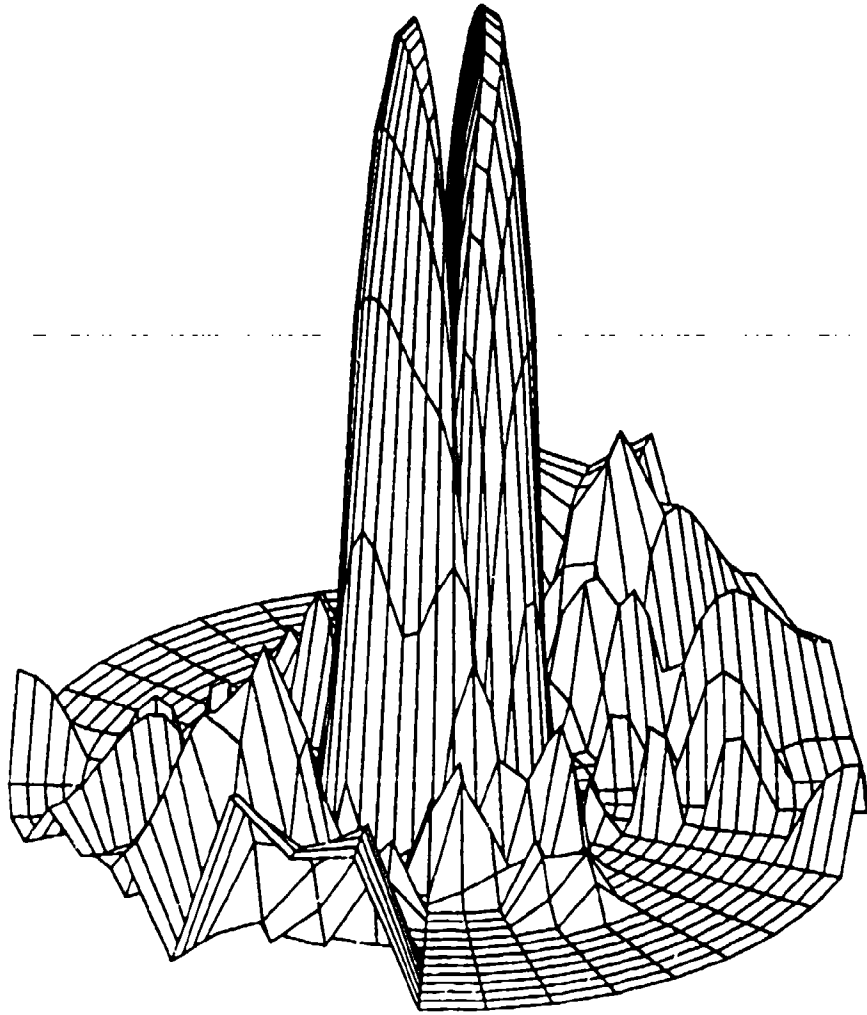


Fig. 11. Difference Power Pattern with 3 Element Failures

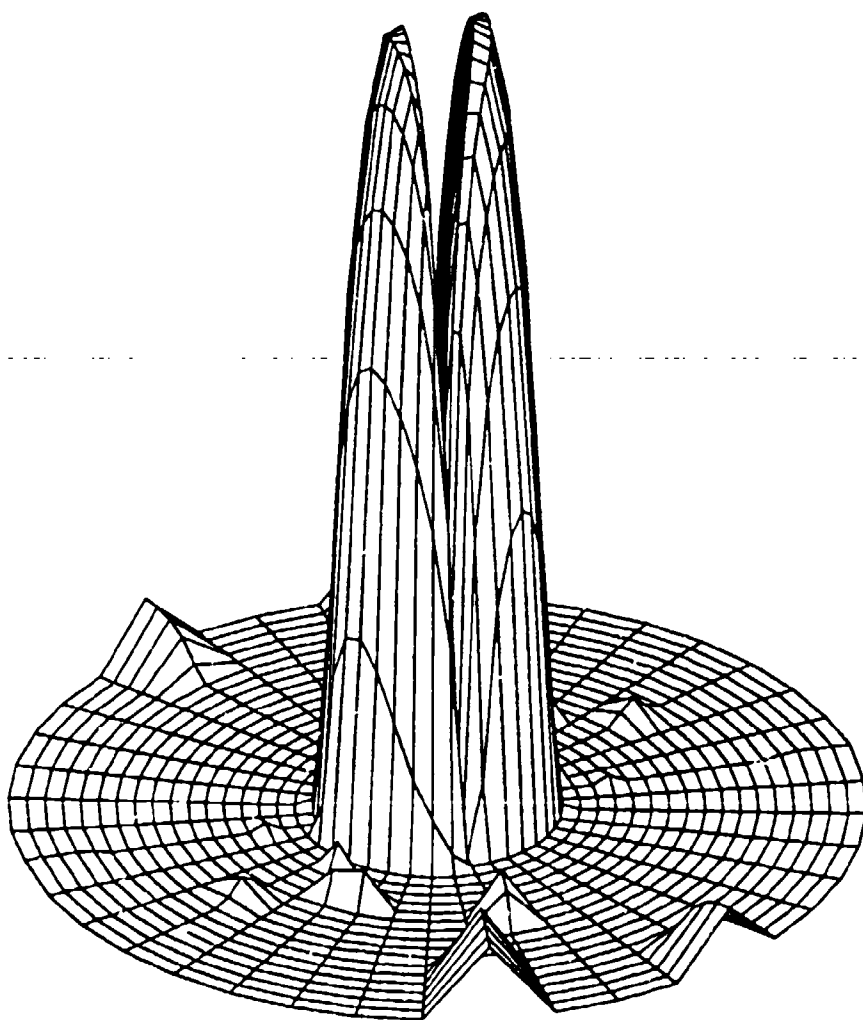


Fig. 12. Synthesized Difference Power Pattern with 3 Element Failures

The peak sidelobe level is -28.7 dB with a gain of 17.1 dB. The sidelobe level of the synthesized pattern cannot be brought down to the level of the design pattern as was the case with the sum pattern, because the degrees of freedom are now about halved. The half-power null width between peaks increased to 9.09°.

The results of this test case and several others, indicate that it is possible to reduce the sidelobe level, of an array with failed elements, down to near the design level. However, the success depends on the nature of the failure. For example, if the central element fails in a sum pattern, the synthesis yields no improvement. This is to be expected since this failure is quite similar to blockage. As expected, elements which fail and have a large relative weight are much more difficult to recover from than elements with a small weight. Also, clustered failures are easier to recover from than the same number of random failures. This is because a clustered failure yields a relatively local increase in the sidelobe region as opposed to the same number of random failures which yields a uniform increase in the sidelobe level. It was interesting to observe that it was easier to recover from an entire row failure than from the same number of random failures. In general, the decrease in the sidelobe level comes with a drop in gain and a slightly broader main beam.

VII. Conclusion

The performance degradation of arrays with element failures may be partially compensated for by a redistribution of the amplitude and phase over the remaining elements. The extent of this compensation is determined by the number and location of the failed elements. The accumulated averaging scheme, combined with the conjugate gradient algorithm, provides a very stable means to synthesize the new array aperture distribution.

References

1. J. F. Deford and O. P. Gandhi, "Phase-Only Synthesis of Minimum Peak Sidelobe Patterns for Linear and Planar Arrays," *IEEE Trans. Antennas Propagat.*, vol. 36, no. 2, pp. 191-201, Feb. 1988.
2. T. J. Peters, "Application of a Conjugate Gradient Method to the Synthesis of Phase-Only Arrays," Aerospace Corp., Los Angeles, CA, Tech. Rep. TR-0090(5925-05)-1, Feb. 1991.
3. M. R. Hestenes and E. Steifel, "Method of Conjugate Gradients for Solving Linear Systems," *J. Res. Nat. Bur. Standard.*, vol. 49, no. 6, pp. 409-436, Dec. 1952.
4. M. R. Hestenes, "Conjugate Direction Methods in Optimization," New York: Springer-Verlag, 1980, pp.135-140.
5. T. J. Peters, "Computation of the Scattering by Planar and Non-Planar Plates Using a Conjugate Gradient FFT Method," Ph.D. dissertation, Radiation Laboratory, University of Michigan, 1988.
6. T. S. Fong and R. A. Birgenheier, "Method of Conjugate Gradients for Antenna Pattern Synthesis," *Radio Sci.*, vol. 6, no. 12, pp. 1123-1130, Dec. 1971.
7. W. A. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, "Conjugate Gradient Methods in Multidimensions," in *Numerical Recipes*: Cambridge Cambridge University Press, 1988, pp. 301-307.

Appendix: Fortran 77 Programs

This appendix contains the Fortran 77 programs `SUMSYN` and `DIFSYN` as well as all associated subroutines. The program `SUMSYN` performs the sum pattern synthesis with element failures and the program `DIFSYN` performs difference pattern synthesis with element failures. The required inputs to each program are listed in the comments found in the source code. These programs were used to generate all the results presented in this report.

```

1      PROGRAM SUMSYN
2 C      *****
3 C      * THIS PROGRAM SYNTHESIZES THE AMPLITUDE AND PHASE DISTRIBUTION *
4 C      * NECESSARY TO GENERATE A SUM PATTERN OVER A HEXAGONAL ARRAY OF *
5 C      * POINT SOURCES WITH SOME ELEMENT FAILURES. *
6 C      *****
7 C      * TIMOTHY J. PETERS                                LAST UPDATED *
8 C      * THE AEROSPACE CORPORATION                        3/1/91 *
9 C      * 2360 EAST EL SEGUNDO BOULEVARD. *
10 C      * EL SEGUNDO, CA 90245 *
11 C      *****
12 C      * -INPUTS:----- *
13 C      * * *
14 C      * NL - NUMBER OF LAYERS FORMING THE HEXAGONAL LATTICE *
15 C      * NTE - NUMBER OF TOTAL ELEMENTS = 1+3*NL*(NL+1) *
16 C      * NBE - NUMBER OF BAD ELEMENTS *
17 C      * BAD(1..NBE) - INTEGER ARRAY HOLDING THE NUMBER POSITION OF EACH *
18 C      * BAD ELEMENT IN THE ARRAY. *
19 C      * DS - THE ELEMENT SPACING IN WAVELENGTHS. *
20 C      * NE - NUMBER OF ACTUAL ELEMENTS = NTE-NBE. *
21 C      * ET - THE EDGE TAPER OF THE PARABOLIC DISTRIBUTION. *
22 C      * N - THE ORDER OF THE PARABOLIC DISTRIBUTION. *
23 C      * BNF - THE THEORETICAL BEAM WIDTH NULL FACTOR FOR THE CHOSEN *
24 C      * DISTRIBUTION. *
25 C      * BBF - THE BEAM WIDTH BROADENING FACTOR. THIS ALLOWS THE BEAM *
26 C      * TO BROADEN BEYOND THE DESIGN VALUE IN ORDER TO REDUCE THE *
27 C      * SIDELOBES WHEN ELEMENTS FAIL. *
28 C      * NSP - NUMBER OF SAMPLE POINTS IN THE SIDELobe REGION. *
29 C      * NS - NUMBER OF RESTARTS. *
30 C      * NI - NUMBER OF CONJUGATE GRADIENT ITERATIONS PER RESTART. *
31 C      * * *
32 C      * OUTPUTS: *
33 C      * * *
34 C      * U(NE) - REAL PART OF THE EXCITATION OF EACH ELEMENT. *
35 C      * V(NE) - IMAGINARY PART OF THE EXCITATION OF EACH ELEMENT. *
36 C      * * *
37 C      *****
38      PARAMETER (NL=5,NTE=91,NBE=3,NE=88,NSP=1650)
39      REAL*4 U(NE),V(NE),I(NE),Y(NE),SU(NE),SV(NE),GU(NE),GV(NE)
40      REAL*4 HU(NE,NE),HUV(NE,NE),HVV(NE,NE),HVV(NE,NE)
41      REAL*4 PU(NE),PV(NE),QU(NE),QV(NE),RU(NE),RV(NE),ZU(NE),ZV(NE)
42      REAL*4 XI(NSP),KY(NSP)
43      INTEGER IP(NSP),BAD(NE)
44      OPEN(UNIT=2,FILE='OUTPUT',STATUS='UNKNOWN')
45      OPEN(UNIT=4,FILE='SUM_FIXED_EST',STATUS='UNKNOWN')
46 C      *****
47 C      * GENERATE THE GEOMETRY. *
48 C      *****
49      DS=0.6

```

```

50      BAD(1)=35
51      BAD(2)=37
52      BAD(3)=44
53      CALL GEOMET(NL,DS,NE,NBE,BAD,X,Y)
54 C *****
55 C * INITIALIZE THE TAPER DISTRIBUTION. *
56 C *****
57      ET=0.1
58      N=2
59      BNF=1.5
60      IFLAG1=0
61      IF (IFLAG1 .EQ. -1) THEN
62          CALL TAPER(NE,U,V,X,Y,ET,N,NL,DS)
63          REWIND 4
64          WRITE(4,101) (I,U(I),V(I),I=1,NE)
65      ELSE
66          END IF
67          REWIND 4
68          READ(4,101) (I,U(I),V(I),I=1,NE)
69 C *****
70 C * COMPUTE THE GAIN (DIRECTIVITY) OF THE ARRAY. *
71 C *****
72      CALL GAIN(NE,U,V,X,Y,GDB)
73 C *****
74 C * GET THE BEAM WIDTH BETWEEN FIRST NULLS. *
75 C *****
76      CALL BEAM(NE,U,V,X,Y,BWFN)
77 C *****
78 C * GENERATE THE K-SPACE SAMPLE POINTS. *
79 C *****
80      BBF=7.0
81      CALL KSPACE(NMP,NL,DS,BNF,BBF,KX,KY)
82 C *****
83 C * COMPUTE POWER IN MAIN BEAM. *
84 C *****
85      CALL POWI(NE,U,V,X,Y,0.0,0.0,PO)
86 C *****
87 C * COMPUTE PEAK SIDELobe POWER. *
88 C *****
89      NP=0
90      CALL PEAKSL(NE,U,V,X,Y,NSP,NMP,KX,KY,IP,NP,PMAI)
91 C *****
92 C * COMPUTE NORMALIZED PEAK SIDELobe POWER IN DB. *
93 C *****
94      PMAIDB=10.0*ALOG10(PMAI/PO)
95 C *****
96 C * PERFORM CONJUGATE GRADIENT STEPS. *
97 C *****
98      NS=300

```

```

99      NI=3
100     CALL CGRAD(NE,NSP,NMP,NS,NI,U,V,I,Y,SU,SV,NP,IP,KX,KY,GU,GV,HUU
101     &,HUV,HVU,HVV,PU,PV,QU,QV,RU,RV,ZU,ZV)
102     REWIND 4
103     WRITE(4,101) (I,U(I),V(I),I=1,NE)
104 C    *****
105 C    * POWER PATTERN GRAPHICS OUTPUT *
106 C    *****
107     REWIND 4
108     READ(4,101) (I,U(I),V(I),I=1,NE)
109 C    *****
110 C    * PRINT THE POWER PATTERN IN CYLINDRICAL COORDINATES. *
111 C    *****
112     CALL POWPAT(NE,U,V,I,Y)
113 C    *****
114 C    * PRINT THE POWER PATTERN IN SPHERICAL COORDINATES. *
115 C    *****
116     CALL SPOWPAT(NE,U,V,I,Y)
117 C    *****
118 C    * FORMATS. *
119 C    *****
120 100   FORMAT(16,1X,F15.5,1X,F15.5)
121 101   FORMAT(16,1X,F15.7,1X,F15.7)
122     END
123 C
124     SUBROUTINE GEOMET(NL,S,NE,NBE,BAD,X,Y)
125 C    *****
126 C    * COMPUTE THE POSITION OF EACH ELEMENT IN A HEXAGONAL ARRAY. *
127 C    *****
128     REAL*4 X(NE),Y(NE)
129     INTEGER BAD(NE),IFLAG
130     HGT=(SQRT(3.0)/2.0)*S
131 C    *****
132 C    * CENTER ROW. *
133 C    *****
134     K=0
135     L=0
136     XMIN=-NL*S
137     DO 1 I=0,2*NL
138     K=K+1
139     L=L+1
140     CALL CHECK(L,NE,NBE,BAD,IFLAG)
141     IF (IFLAG .EQ. 1) THEN
142     K=K-1
143     ELSE
144     X(K)=XMIN+I*S
145     Y(K)=0.0
146     END IF
147 1     CONTINUE

```

```

148 C *****
149 C * TOP ROWS. *
150 C *****
151     INUM=2*NL+1
152     DO 2 J=1,NL
153         INUM=INUM-1
154         IMIN=-NL*S+J*S/2.0
155         DO 3 I=0,INUM-1
156             K=K+1
157             L=L+1
158             CALL CHECK(L,NE,NBE,BAD,IFLAG)
159             IF (IFLAG .EQ. 1) THEN
160                 K=K-1
161             ELSE
162                 X(K)=IMIN+I*S
163                 Y(K)=HGT*J
164             END IF
165 3     CONTINUE
166 2     CONTINUE
167 C *****
168 C * BOTTOM ROWS. *
169 C *****
170     INUM=2*NL+1
171     DO 4 J=1,NL
172         INUM=INUM-1
173         IMIN=-NL*S+J*S/2.0
174         DO 5 I=0,INUM-1
175             K=K+1
176             L=L+1
177             CALL CHECK(L,NE,NBE,BAD,IFLAG)
178             IF (IFLAG .EQ. 1) THEN
179                 K=K-1
180             ELSE
181                 X(K)=IMIN+I*S
182                 Y(K)=-HGT*J
183             END IF
184 5     CONTINUE
185 4     CONTINUE
186     RETURN
187     END
188 C
189     SUBROUTINE CHECK(L,NE,NBE,BAD,IFLAG)
190     INTEGER BAD(NE)
191     IFLAG=0
192     DO 1 I=1,NBE
193         IF (L .EQ. BAD(I)) THEN
194             IFLAG=1
195         ELSE
196             IFLAG=0
197         END IF
198     END DO
199     END

```

```

197 1    CONTINUE
198      RETURN
199      END
200 C
201      SUBROUTINE BEAM(NE,U,V,X,Y,BWFN)
202 C      *****
203 C      * COMPUTE THE BEAM WIDTH BETWEEN FIRST NULLS. *
204 C      *****
205      REAL U(NE),V(NE),X(NE),Y(NE),KXI,KYI,KIP,KYP
206      RAD=.17453293E-01
207      TP=.6283185E+01
208      TMIN=0.0
209      TMAX=10.0
210      P=0.0
211      NT=140
212      DT=(TMAX-TMIN)/NT
213 C      *****
214 C      * FIND THE MAXIMUM VALUE. *
215 C      *****
216      CALL POWER(NE,U,V,X,Y,0.0,0.0,PEAK)
217 C      *****
218 C      * RECOMPUTE AND NOW FIND THE POINT WHICH IS AT THE SLL BELOW THE *
219 C      * PEAK. *
220 C      *****
221      DO 1 K=1,90
222          P=P+1.0
223          DO 2 I=0,NT
224              T=TMIN+I*DT
225              TPS=TP*SIN(RAD*T)
226              KXI=TPS*COS(RAD*P)
227              KYI=TPS*SIN(RAD*P)
228              CALL POWI(NE,U,V,X,Y,KXI,KYI,POW)
229              IF (POW/PEAK .LE. 0.5) THEN
230                  WRITE(2,100) T,10.0*ALOG(POW/PEAK)
231                  GO TO 99
232              ELSE
233                  END IF
234 2          CONTINUE
235 99      CONTINUE
236      BWFN=2.0*T
237      WRITE(*,*) 'P BWFN ',P,BWFN
238 1      CONTINUE
239 100     FORMAT(F10.5,1X,F10.5)
240      RETURN
241      END
242 C
243      SUBROUTINE KSPACE(NMP,NL,DS,BNF,BBF,KI,KY)
244 C      *****
245 C      * THIS SUBROUTINE SAMPLES THE KI >0 REGION OF K SPACE. *

```



```

246 C *****
247 REAL*4 KI(*),KY(*),KC,KR,KIX,KYY,KIMIN,KXMAX,KYMIN,KYMAX
248 RAD=.17453293E-01
249 PI=.3141593E+01
250 TP=.0283185E+01
251 C *****
252 C * COMPUTE THE APPROXIMATE DIAMETER OF THE APERTURE. *
253 C *****
254 D=2.0*DS*PL
255 C *****
256 C * COMPUTE THE BEAM WIDTH BETWEEN FIRST NULLS OF THE DESIGN *
257 C * PATTERN. *
258 C *****
259 BWFN=BNF*2.439272/D
260 C *****
261 C * COMPUTE THE SMALLEST BEAM WIDTH BETWEEN FIRST NULLS ALLOWED BY *
262 C * THE SYNTHESIS ALGORITHM. *
263 C *****
264 BWFNS=BBF*BWFN
265 C *****
266 C * COMPUTE THE K SPACE RADIUS OF THE ALLOWED MAIN BEAM. *
267 C *****
268 KC=TP*0.34
269 KC=BWFNS/2.0
270 C *****
271 C * GENFRATE THE RECTANGULAR LATTICE OF SAMPLE POINTS. *
272 C *****
273 KYMIN=-TP
274 KYMAX=TP
275 KIMIN=-TP
276 KXMAX=TP
277 NKX=45
278 NKY=45
279 DKX=(KXMAX-KIMIN)/(NKX-1)
280 DKY=(KYMAX-KYMIN)/(NKY-1)
281 NMP=0
282 DO 1 I=0,NKY
283 KYY=KYMIN+I*DKY
284 DO 2 J=0,NKX
285 XXX=KIMIN+J*DKX
286 KR=SQRT(KIX*KIX+KYY*KYY)
287 IF ((KR.GT. KC) .AND. (KR.LT. TP)) THEN
288 NMP=NMP+1
289 KX(NMP)=XXX
290 KY(NMP)=KYY
291 ELSE
292 END IF
293 2 CONTINUE
294 1 CONTINUE

```

```

295      RETURN
296      END
297 C
298      SUBROUTINE TAPFR(NE,U,V,X,Y,ET,N,NL,S)
299      REAL U(NE),V(NE),X(NE),Y(NE)
300 C      *****
301 C      * COMPUTE APPROXIMATE ARRAY RADIUS.
302 C      *****
303      A=S*NL
304 C      *****
305 C      * COMPUTE AMPLITUDE.
306 C      *****
307      DO 1 K=1,NE
308          R=SQRT(X(K)*X(K)+Y(K)*Y(K))
309          AMP=ET*(1.0-ET)*((1.0-(R/A)**2)**N)
310          U(K)=AMP
311          V(K)=0.0
312 1      CONTINUE
313      RETURN
314      END
315 C
316      SUBROUTINE CGRAD(NE,NSP,NMP,NS,N1,U,V,X,Y,SU,SV,NP,IP,KI,KY,GU,GV
317      ,HUU,HUV,HVU,HVV,PJ,PV,QU,QV,RU,RV,ZU,ZV)
318 C      *****
319 C      * COMPUTE POWER IN A SINGLE DIRECTION.
320 C      *****
321      REAL*4 U(NE),V(NE),X(NE),Y(NE),SU(NE),SV(NE),GU(NE),GV(NE)
322      REAL*4 HUU(NE,NE),HUV(NE,NE),HVU(NE,NE),HVV(NE,NE)
323      REAL*4 PU(NE),PV(NE),QU(NE),QV(NE),RU(NE),RV(NE),ZU(NE),ZV(NE)
324      REAL*4 KI(NSP),KY(NSP)
325      REAL*4 KXI,KYI
326      INTEGER IP(NSP)
327 C      *****
328 C      * INITIALIZE THE NUMBER OF SIDELobe SAMPLE POINTS.
329 C      *****
330      NP=0
331 C      *****
332 C      * PERFORM NS RE-STARTS.
333 C      *****
334      DO 1 L=1,NS
335 C      *****
336 C      * FIND THE PEAK SIDELobe POWER POINT AND UPDATE THE NUMBER
337 C      * OF MATCH POINTS.
338 C      *****
339      CALL PEAKSL(NE,U,V,X,Y,NSP,NMP,KI,KY,IP,NP,PMAX)
340      WRITE(*,160) (IP(I),I=1,NP)
341      WRITE(2,160) (IP(I),I=1,NP)
342 160      FORMAT(10I5)
343 C      *****

```

```

344 C      * FIND THE POWER IN THE MAIN BEAM.      *
345 C      *****
346      KXI=0.0
347      KYI=0.0
348      CALL POWI(NE,U,V,X,Y,KXI,KYI,PO)
349 C      *****
350 C      * COMPUTE AND STORE AN ARRAY OF THE 1ST DERIVATIVE OF THE *
351 C      * POWER IN THE MAIN BEAM DIRECTION WITH RESPECT TO THE MTH *
352 C      * VARIABLE.      *
353 C      *****
354      KXI=0.0
355      KYI=0.0
356      DO 2 M=1,NE
357          CALL DPOWIU(NE,U,V,X,Y,M,KXI,KYI,DPWIU)
358          SU(M)=DPWIU
359          CALL DPOWIV(NE,U,V,X,Y,M,KXI,KYI,DPWIV)
360          SV(M)=DPWIV
361 2      CONTINUE
362 C      *****
363 C      * COMPUTE AND STORE THE AVERAGE SIDELOBE POWER.      *
364 C      *****
365      PA=0.0
366      DO 3 I=1,NP
367          CALL POWI(NE,U,V,X,Y,KX(IP(I)),KY(IP(I)),POWERI)
368          PA=PA+POWERI
369 3      CONTINUE
370      PA=PA/NP
371      WRITE(*,200) L,10.0*ALOG10(PA/PO),10.0*ALOG10(PMAX/PO)
372      WRITE(2,200) L,10.0*ALOG10(PA/PO),10.0*ALOG10(PMAX/PO)
373 200      FORMAT(1X,I5,1X,'AVE PEAK SLL',1X,F15.6,1X,
374          &      'PEAK SLL',1X,F15.6)
375 C      *****
376 C      * COMPUTE THE NEGATIVE OF THE GRADIENT.      *
377 C      *****
378      DO 4 M=1,NE
379 C      *****
380 C      * COMPUTE THE DERIVATIVE OF THE AVERAGE SIDELOBE POWER. *
381 C      *****
382      DPAMU=0.0
383      DPAMV=0.0
384      DO 5 I=1,NP
385          CALL DPOWIU(NE,U,V,X,Y,M,KX(IP(I)),KY(IP(I)),DPWIU)
386          DPAMU=DPAMU+DPWIU
387          CALL DPOWIV(NE,U,V,X,Y,M,KX(IP(I)),KY(IP(I)),DPWIV)
388          DPAMV=DPAMV+DPWIV
389 5      CONTINUE
390      DPAMU=DPAMU/NP
391      DPAMV=DPAMV/NP
392 C      *****

```

```

393 C          * COMPUTE THE NEGATIVE GRADIENT.          *
394 C          *****
395             GU(M)=(PA*SU(M)/PO-DPAMU)/PO
396             GV(M)=(PA*SV(M)/PO-DPAMV)/PO
397 4          CONTINUE
398 C          *****
399 C          * COMPUTE THE SYMMETRIC HESSIAN SUB MATRICES HUU AND HVV *
400 C          *****
401             DO 6 M=1,NE
402             DO 7 N=M,NE
403                 KXI=0.0
404                 KYI=0.0
405                 CALL DDPIUU(NE,X,Y,M,N,KXI,KYI,DDPUU)
406                 DDPOUU=DDPUU
407                 CALL DDPIVV(NE,X,Y,M,N,KXI,KYI,DDPVV)
408                 DDPOVV=DDPVV
409                 DDPAUU=0.0
410                 DDPAVV=0.0
411                 DO 8 I=1,NP
412                     CALL DDPIUU(NE,X,Y,M,N,KXI(IP(I)),KYI(IP(I)),DDPUU)
413                     DDPAUU=DDPAUU+DDPUU
414                     CALL DDPIVV(NE,X,Y,M,N,KXI(IP(I)),KYI(IP(I)),DDPVV)
415                     DDPAVV=DDPAVV+DDPVV
416 8             CONTINUE
417                 DDPAUU=DDPAUU/NP
418                 DDPAVV=DDPAVV/NP
419                 HUU(M,N)=(GU(N)*SU(M)+GU(M)*SU(N)+DDPAUU-PA*DDPOUU/PO)/PO
420                 HUU(N,M)=HUU(M,N)
421                 HVV(M,N)=(GV(N)*SV(M)+GV(M)*SV(N)+DDPAVV-PA*DDPOVV/PO)/PO
422                 HVV(N,M)=HVV(M,N)
423 7             CONTINUE
424 6          CONTINUE
425 C          *****
426 C          * COMPUTE THE ASYMMETRIC HESSIAN SUB MATRICE HUV.          *
427 C          *****
428             DO 9 M=1,NE
429             DO 10 N=1,NE
430                 KXI=0.0
431                 KYI=0.0
432                 CALL DDPIUV(NE,X,Y,M,N,KXI,KYI,DDPUV)
433                 DDPOUV=DDPUV
434                 DDPAUV=0.0
435                 DO 11 I=1,NP
436                     CALL DDPIUV(NE,X,Y,M,N,KXI(IP(I)),KYI(IP(I)),DDPUV)
437                     DDPAUV=DDPAUV+DDPUV
438 11            CONTINUE
439                 DDPAUV=DDPAUV/NP
440                 HUV(M,N)=(GV(N)*SU(M)+GU(M)*SV(N)+DDPAUV-PA*DDPOUV/PO)/PO
441                 HVU(N,M)=HUV(M,N)

```

```

442 10      CONTINUE
443 9       CONTINUE
444 C      *****
445 C      * START THE CONJUGATE GRADIENT ALGORITHM. *
446 C      *****
447 C      *****
448 C      * INITIALIZE THE RESIDUAL. *
449 C      *****
450          DO 12 M=1,NE
451              RU(M)=GU(M)
452              RV(M)=GV(M)
453              ZU(M)=0.0
454              ZV(M)=0.0
455 12      CONTINUE
456 C      *****
457 C      * INITIALIZE SEARCH VECTOR. *
458 C      *****
459          CALL MATVEC(NE,HUU,RU,HUV,RV,QU)
460          CALL MATVEC(NE,HVU,RU,HVV,RV,QV)
461          CALL NORM22(NE,QU,QUN)
462          CALL NORM22(NE,QV,QVN)
463          BEO=1.0/(QUN+QVN)
464          DO 13 I=1,NE
465              PU(I)=BEO*QU(I)
466              PV(I)=BEO*QV(I)
467 13      CONTINUE
468 C      *****
469 C      * PERFORM CONJUGATE GRADIENT ITERATIONS. *
470 C      *****
471          DO 14 K=1,NI
472 C      *****
473 C      * UPDATE AMPLITUDE VECTOR AND RESIDUAL. *
474 C      *****
475          CALL MATVEC(NE,HUU,PU,HUV,PV,QU)
476          CALL MATVEC(NE,HVU,PU,HVV,PV,QV)
477          CALL NORM22(NE,QU,QUN)
478          CALL NORM22(NE,QV,QVN)
479          AK=1.0/(QUN+QVN)
480          DO 15 I=1,NE
481              ZU(I)=ZU(I)+AK*PU(I)
482              ZV(I)=ZV(I)+AK*PV(I)
483              RU(I)=RU(I)-AK*QU(I)
484              RV(I)=RV(I)-AK*QV(I)
485 15      CONTINUE
486          CALL NORM22(NE,RU,RUNM)
487          CALL NORM22(NE,RV,RVNM)
488          ERR=SQRT(RUNM+RVNM)
489          WRITE(*,500) L,K,ERR
490          WRITE(2,500) L,K,ERR

```

```

491 500      FORMAT(3X,'RESIDUAL',1X,I3,1X,I3,1X,E15.8)
492 C      *****
493 C      * UPDATE SEARCH VECTOR. *
494 C      *****
495      CALL MATVEC(NE,HUU,RU,HUV,RV,QU)
496      CALL MATVEC(NE,HVU,RU,HVV,RV,QV)
497      CALL NORM22(NE,QU,QUN)
498      CALL NORM22(NE,QV,QVN)
499      BEK=1.0/(QUN+QVN)
500      DO 16 I=1,NE
501          PU(I)=PU(I)+BEK*QU(I)
502          PV(I)=PV(I)+BEK*QV(I)
503      16   CONTINUE
504      14   CONTINUE
505 C      *****
506 C      * UPDATE AMPLITUDE VECTOR. *
507 C      *****
508      DO 17 I=1,NE
509          U(I)=U(I)+ZU(I)
510          V(I)=V(I)+ZV(I)
511      17   CONTINUE
512      CALL NORMAL(NE,U,V)
513 C      *****
514 C      * WRITE OUT THE NEW ESTIMATE TO A FILE. *
515 C      *****
516      REWIND 4
517      WRITE(4,101) (I,U(I),V(I),I=1,NE)
518 101  FORMAT(I5,1X,F15.7,1X,F15.7)
519      IF (NP .EQ. NMP) THEN
520          GO TO 99
521      ELSE
522          END IF
523      1     CONTINUE
524 99     CONTINUE
525      RETURN
526      END
527 C
528      SUBROUTINE PEAKSL(NE,U,V,I,Y,NSP,NMP,KI,KY,IP,NP,PMAX)
529 C      *****
530 C      * THIS SUBROUTINE FINDS THE PEAK SIDELobe POWER LEVEL COORDINATE *
531 C      * POINT (KI(LMAX),KY(LMAX)). IF THE POINT IS NEW THEN IT IS *
532 C      * ACCUMULATED INTO THE PCINTER ARRAY IP. THE NUMBER OF POINTS IN *
533 C      * THE INTEGER POINTER ARRAY IP IS NP SUCH THAT IP(I),I=1...NP. *
534 C      *****
535      REAL U(NE),V(NE),X(NE),Y(NE),KI(NSP),KY(NSP)
536      INTEGER IP(NSP)
537 C      *****
538 C      * COMPUTE THE POWER AT EACH POINT AND FIND THE MAX. *
539 C      *****

```

```

540      PMAI=-1000.0
541      LMAI=-1
542      DO 1 I=1,NMP
543          CALL POWI(NE,U,V,X,Y,KX(I),KY(I),PWI)
544          IF (PWI .GT. PMAI) THEN
545              LMAI=I
546              PMAI=PWI
547          ELSE
548              END IF
549 1      CONTINUE
550 C      *****
551 C      * COMPARE LMAI TO ALL THE PREVIOUS POINTERS IP(I),I=1...NP. *
552 C      * IF LMAI MATCHES ONE OF THE PREVIOUS POINTERS THEN THE *
553 C      * POINTER ARRAY NEED NOT BE UPDATED (IFLAG=1). IF LMAI DOES *
554 C      * NOT MATCH A PREVIOUS POINTER VALUE (IFLAG=0) THEN IT *
555 C      *****
556      IFLAG=0
557      DO 2 I=1,NP
558          IF (IP(I) .EQ. LMAI) THEN
559              IFLAG=1
560          ELSE
561              END IF
562 2      CONTINUE
563      IF (IFLAG .EQ. 0) THEN
564          NP=NP+1
565          IP(NP)=LMAI
566      ELSE
567          END IF
568      RETURN
569      END
570 C
571      SUBROUTINE POWI(NE,U,V,X,Y,KXI,KYI,POWERI)
572 C      *****
573 C      * COMPUTE POWER IN A SINGLE DIRECTION. *
574 C      *****
575      REAL U(NE),V(NE),X(NE),Y(NE),KXI,KYI
576      SUM1=0.0
577      SUM2=0.0
578      DO 1 K=1,NE
579          PSI=KXI*X(K)+KYI*Y(K)
580          CP=COS(PSI)
581          SP=SIN(PSI)
582          SUM1=SUM1+U(K)*CP-V(K)*SP
583          SUM2=SUM2+U(K)*SP+V(K)*CP
584 1      CONTINUE
585      TI=SUM1*SUM1
586      VI=SUM2*SUM2
587      POWERI=TI+VI
588      RETURN

```

```

589      END
590 C
591      SUBROUTINE DPOWIU(NE,U,V,X,Y,M,KXI,KYI,DPOWIU)
592 C      *****
593 C      * COMPUTE 1ST DERIVATIVE OF POWER WITH RESPECT TO THE MTH VARIABLE*
594 C      * IN THE ITH DIRECTION. *
595 C      *****
596      REAL U(NE),V(NE),X(NE),Y(NE),KXI,KYI
597 C      *****
598 C      * COMPUTE SUMS. *
599 C      *****
600      SUM1=0.0
601      SUM2=0.0
602      DO 1 K=1,NE
603          PSI=KXI*X(K)+KYI*Y(K)
604          CP=COS(PSI)
605          SP=SIN(PSI)
606          SUM1=SUM1+U(K)*CP-V(K)*SP
607          SUM2=SUM2+U(K)*SP+V(K)*CP
608 1      CONTINUE
609 C      *****
610 C      * COMPUTE 1ST DERIVATIVE OF POWER. *
611 C      *****
612      PSIM=KXI*X(M)+KYI*Y(M)
613      DPOWIU=2.0*(COS(PSIM)*SUM1+SIN(PSIM)*SUM2)
614      RETURN
615      END
616 C
617      SUBROUTINE DPOWIV(NE,U,V,X,Y,M,KXI,KYI,DPOWIV)
618 C      *****
619 C      * COMPUTE 1ST DERIVATIVE OF POWER WITH RESPECT TO THE MTH VARIABLE*
620 C      * IN THE ITH DIRECTION. *
621 C      *****
622      REAL U(NE),V(NE),X(NE),Y(NE),KXI,KYI
623 C      *****
624 C      * COMPUTE SUMS. *
625 C      *****
626      SUM1=0.0
627      SUM2=0.0
628      DO 1 K=1,NE
629          PSI=KXI*X(K)+KYI*Y(K)
630          CP=COS(PSI)
631          SP=SIN(PSI)
632          SUM1=SUM1+U(K)*CP-V(K)*SP
633          SUM2=SUM2+U(K)*SP+V(K)*CP
634 1      CONTINUE
635 C      *****
636 C      * COMPUTE 1ST DERIVATIVE OF POWER. *
637 C      *****

```



```

038      PSIM=KXI*X(M)+KYI*Y(M)
039      DPWIV=2.0*(COS(PSIM)*SUM2-SIN(PSIM)*SUM1)
040      RETURN
041      END
042 C
043      SUBROUTINE DDPIUV(NE,X,Y,M,N,KXI,KYI,DDPUV)
044 C      *****
045 C      * COMPUTE 2ND DERIVATIVE OF POWER WITH RESPECT TO THE M AND NTH *
046 C      * VARIABLES IN THE ITH DIRECTION. *
047 C      *****
048      REAL*4 X(NE),Y(NE),KXI,KYI
049 C      *****
050 C      * COMPUTE DIFFERENCE OF PSI FUNCTIONS. *
051 C      *****
052      PSIM=KXI*X(M)+KYI*Y(M)
053      PSIN=KXI*X(N)+KYI*Y(N)
054      DIF=PSIM-PSIN
055 C      *****
056 C      * COMPUTE 2ND DERIVATIVE OF POWER. *
057 C      *****
058      DDPUV=2.0*COS(DIF)
059      RETURN
060      END
061 C
062      SUBROUTINE DDPIUV(NE,X,Y,M,N,KXI,KYI,DDPUV)
063 C      *****
064 C      * COMPUTE 2ND DERIVATIVE OF POWER WITH RESPECT TO THE M AND NTH *
065 C      * VARIABLES IN THE ITH DIRECTION. *
066 C      *****
067      REAL*4 X(NE),Y(NE),KXI,KYI
068 C      *****
069 C      * COMPUTE DIFFERENCE OF PSI FUNCTIONS. *
070 C      *****
071      PSIM=KXI*X(M)+KYI*Y(M)
072      PSIN=KXI*X(N)+KYI*Y(N)
073      DIF=PSIM-PSIN
074 C      *****
075 C      * COMPUTE 2ND DERIVATIVE OF POWER. *
076 C      *****
077      DDPUV=2.0*SIN(DIF)
078      RETURN
079      END
080 C
081      SUBROUTINE DDPIVV(NE,X,Y,M,N,KXI,KYI,DDPVV)
082 C      *****
083 C      * COMPUTE 2ND DERIVATIVE OF POWER WITH RESPECT TO THE M AND NTH *
084 C      * VARIABLES IN THE ITH DIRECTION. *
085 C      *****
086      REAL*4 X(NE),Y(NE),KXI,KYI

```

```

687 C *****
688 C * COMPUTE DIFFERENCE OF PSI FUNCTIONS. *
689 C *****
690 PSIM=KXI+X(M)+KYI+Y(M)
691 PSIN=KXI+X(N)+KYI+Y(N)
692 DIF=PSIM-PSIN
693 C *****
694 C * COMPUTE 2ND DERIVATIVE OF POWER. *
695 C *****
696 DDPVV=2.0+COS(DIF)
697 RETURN
698 END
699 C
700 SUBROUTINE NORM22(N,A,AN)
701 C *****
702 C * THIS SUBROUTINE COMPUTES THE EUCLIDIAN NORM SQUARED OF A. *
703 C *****
704 REAL*4 A(N)
705 AN=0.0
706 DO 1 I=1,N
707 AN=AN+A(I)*A(I)
708 1 CONTINUE
709 RETURN
710 END
711 C
712 SUBROUTINE MATVEC(N,A,B,C,D,E)
713 C *****
714 C * THIS SUBROUTINE COMPUTES THE MATRIX PRODUCT E=AB+CD. *
715 C *****
716 REAL*4 A(N,N),B(N),C(N,N),D(N),E(N)
717 DO 1 I=1,N
718 SUM=0.0
719 DO 2 J=1,N
720 SUM=SUM+A(I,J)*B(J)+C(I,J)*D(J)
721 2 CONTINUE
722 E(I)=SUM
723 1 CONTINUE
724 RETURN
725 END
726 C
727 SUBROUTINE NORMAL(N,A,B)
728 C *****
729 C * THIS SUBROUTINE NORMALIZES A VECTOR OF LENGTH N. *
730 C *****
731 REAL*4 A(N),B(N)
732 VMAX=SQRT(A(1)*A(1)+B(1)*B(1))
733 DO 1 I=1,N
734 V=SQRT(A(I)*A(I)+B(I)*B(I))
735 IF (V.GT.VMAX) THEN

```

```

736         VMAX=V
737         ELSE
738         END IF
739 1      CONTINUE
740         DO 2 I=1,N
741             A(I)=A(I)/ABS(VMAX)
742             B(I)=B(I)/ABS(VMAX)
743 2      CONTINUE
744         RETURN
745     END
746 C
747     SUBROUTINE GAIN(NE,U,V,I,Y,GDB)
748 C *****
749 C * COMPUTE POWER IN A SINGLE DIRECTION. *
750 C *****
751     REAL U(NE),V(NE),I(NE),Y(NE)
752     RAD=.17453293E-01
753     PI=.3141593E+01
754     TF .0283185E+01
755 C *****
756 C * GRAPH INPUTS *
757 C *****
758     NP=41
759     NR=31
760     PMIN=0.0
761     PMAX=360.0
762     TMIN=0.0
763     TMAX=90.0
764     DP=(PMAX-PMIN)/(NP-1)
765     DR=(TMAX-TMIN)/(NR-1)
766 C *****
767 C * COMPUTE PEAK VALUE. *
768 C *****
769     CALL POWER(NE,U,V,I,Y,0.0,0.0,PEAK)
770 C *****
771 C * COMPUTE THE SOLID ANGLE. *
772 C *****
773     SUM=0.0
774     DO 1 I=0,NR-2
775         T=TMIN+DR/2.0+I*DR
776         ST=SIN(RAD*T)
777         DO 2 J=0,NP-2
778             P=PMIN+DP/2.0+J*DP
779             CALL POWER(NE,U,V,I,Y,T,P,POW)
780             SUM=SUM+ST*POW
781 2      CONTINUE
782 1      CONTINUE
783     TEST=RAD*RAD*DR*DP*SUM
784     SOLID=RAD*RAD*DR*DP*SUM/PEAK

```

```

786         GDB=10.0*ALOG10(TP/SOLID)
786     RETURN
787     END
788 C
788     SUBROUTINE POWPAT(NE,U,V,I,Y)
790 C     *****
791 C     * COMPUTE POWER IN A SINGLE DIRECTION. *
792 C     *****
793     REAL U(NE),V(NE),I(NE),Y(NE)
794     OPEN(UNIT=7,FILE='DATA')
795     RAD=.17453293E-01
796     PI=.3141593E+01
797     TP=.6283185E+01
798 C     *****
799 C     * GRAPH INPUTS *
800 C     *****
801     NP=55
802     NR=30
803     NS=0
804     PMIN=0.0
805     PMAX=360.0
806     TMIN=0.0
807     TMAX=90.0
808     DP=(PMAX-PMIN)/(NP-1)
809     DR=(TMAX-TMIN)/(NR-1)
810     AX=150.0
811     AY=AX
812     AZ=680.0
813     P0=55.0
814     Q0=60.0
815     XA=-2.0
816     XB=2.0
817     YA=-2.0
818     YB=2.0
819     ZA=-2.0
820     ZB=2.3
821     AT=.3
822     WRITE(7,*) NP,NR,NS,AX,AY,AZ
823     WRITE(7,*) XA,XB,YA,YB,ZA,ZB
824     WRITE(7,*) AT,P0,Q0
825 C     *****
826 C     * PHI VALUES. *
827 C     *****
828     DO 1 I=0,NP-1
829         P=PMIN+I*DP
830         WRITE(7,*) RAD*P
831 1     CONTINUE
832 C     *****
833 C     * RADIAL (THETA) VALUES. *

```

```

834 C *****
835     DO 2 I=0,NR-1
836         T=TMIN+I*DR
837         WRITE(7,*) RAD*T
838     2    CONTINUE
839 C *****
840 C     * FUNCTION VALUES.
841 C *****
842 C *****
843 C     * COMPUTE PEAK VALUE.
844 C *****
845         CALL POWER(NE,U,V,I,Y,0.0,0.0,PEAK)
846 C *****
847 C     * COMPUTE EACH VALUE.
848 C *****
849     DO 3 I=0,NR-1
850         T=TMIN+I*DR
851         DO 4 J=0,NP-1
852             P=PMIN+J*DP
853             CALL POWER(NE,U,V,I,Y,T,P,POW)
854             WRITE(7,*) POW/PEAK
855     4    CONTINUE
856 3    CONTINUE
857     RETURN
858     END
859 C
860     SUBROUTINE SPOWPAT(NE,U,V,I,Y)
861 C *****
862 C     * COMPUTE POWER IN A SINGLE DIRECTION.
863 C *****
864     REAL U(NE),V(NE),I(NE),Y(NE)
865     OPEN(UNIT=8,FILE='SDATA')
866     RAD=.17453293E+01
867     PI=.3141593E+01
868     TP=.6283185E+01
869 C *****
870 C     * GRAPH INPUTS
871 C *****
872     NP=115
873     NR=63
874     NS=0
875     PMIN=0.0
876     PMAX=360.0
877     TMIN=0.0
878     TMAX=90.0
879     DP=(PMAX-PMIN)/(NP-1)
880     DR=(TMAX-TMIN)/(NR-1)
881     AX=600.0
882     AY=AX

```

```

883      AZ=AX
884      PO=25.0
885      QO=60.0
886      IA=-2.0
887      IB=2.0
888      YA=-2.0
889      YB=2.0
890      ZA=-2.0
891      ZB=2.3
892      AT=.3
893      IT=4
894      WRITE(8,*) NP,NR,NS,IT,AX,AY,AZ
895      WRITE(8,*) IA,IB,YA,YB,ZA,ZB
896      WRITE(8,*) AT,PO,QO
897 C *****
898 C * PHI VALUES. *
899 C *****
900      DO 1 I=0,NP-1
901          P=PMIN+I*DP
902          WRITE(8,*) RAD*P
903 1      CONTINUE
904 C *****
905 C * RADIAL (THETA) VALUES. *
906 C *****
907      DO 2 I=0,NR-1
908          T=TMIN+I*DR
909          WRITE(8,*) RAD*T
910 2      CONTINUE
911 C *****
912 C * FUNCTION VALUES. *
913 C *****
914 C *****
915 C * COMPUTE PEAK VALUE. *
916 C *****
917      CALL POWER(NE,U,V,X,Y,0.0,0.0,PEAK)
918 C *****
919 C * COMPUTE EACH VALUE. *
920 C *****
921      DO 3 I=0,NR-1
922          T=TMIN+I*DR
923          DO 4 J=0,NP-1
924              P=PMIN+J*DP
925              CALL POWER(NE,U,V,X,Y,T,P,POW)
926              WRITE(8,*) POW/PEAK
927 4          CONTINUE
928 3      CONTINUE
929      RETURN
930      END
931 C

```

```

032      SUBROUTINE POWER(NE,U,V,X,Y,T,P,POW)
033 C      *****
034 C      * COMPUTE POWER IN A SINGLE DIRECTION GIVEN A THETA AND PHI. *
035 C      *****
036      REAL U(NE),V(NE),X(NE),Y(NE),KXI,KYI
037      RAD=.17453293E-01
038      TP=.6283166E+01
039      TPS=TP*SIN(RAD*T)
040      RP=RAD*P
041      KXI=TPS*COS(RP)
042      KYI=TPS*SIN(RP)
043      SUM1=0.0
044      SUM2=0.0
      DO 1 K=1,NE
046          PSI=KXI*X(K)+KYI*Y(K)
047          CP=COS(PSI)
048          SP=SIN(PSI)
049          SUM1=SUM1+U(K)*CP-V(K)*SP
050          SUM2=SUM2+U(K)*SP+V(K)*CP
051 1      CONTINUE
052      TI=SUM1*SUM1
053      VI=SUM2*SUM2
054      POW=TI+VI
055      RETURN
056      END

```

```

1      PROGRAM DIFSYN
2 C    *****
3 C    * THIS PROGRAM SYNTHESIZES THE AMPLITUDE AND PHASE DISTRIBUTION *
4 C    * NECESSARY TO GENERATE A DIFFERENCE PATTERN OVER A HEXAGONAL *
5 C    * ARRAY OF POINT SOURCES WITH SOME ELEMENT FAILURES. *
6 C    *****
7 C    * TIMOTHY J. PETERS                                LAST UPDATED *
8 C    * THE AEROSPACE CORPORATION                        . /91 *
9 C    * 2360 EAST EL SEGUNDO BOULEVARD. *
10 C    * EL SEGUNDO, CA 90246 *
11 C    *****
12 C    * INPUTS:
13 C    *
14 C    * NL - NUMBER OF LAYERS FORMING THE HEXAGONAL LATTICE *
15 C    * NTE - NUMBER OF TOTAL ELEMENTS =(1+3*NL*(NL+1)-NL)/2 FOR NL ODD *
16 C    *                                     =(3*NL*(NL+1)-NL)/2 FOR NL EVEN *
17 C    * NBE - NUMBER OF BAD ELEMENTS *
18 C    * BAD(1..NBE) - INTEGER ARRAY HOLDING THE NUMBER POSITION OF EACH *
19 C    * BAD ELEMENT IN THE ARRAY. *
20 C    * DS - THE ELEMENT SPACING IN WAVELENGTHS. *
21 C    * NE - NUMBER OF ACTUAL ELEMENTS = NTE-NBE. *
22 C    * SLL - SIDELobe LEVEL OF DESIGN BAYLISS DISTRIBUTION. *
23 C    * NBAR - PARAMETER FOR BAYLISS DISTRIBUTION. *
24 C    * BBF - THE BEAM WIDTH BROADENING FACTOR. THIS ALLOWS THE BEAM *
25 C    * TO BROADEN BEYOND THE DESIGN VALUE IN ORDER TO REDUCE THE *
26 C    * SIDELOBES WHEN ELEMENTS FAIL. *
27 C    * NSP - NUMBER OF SAMPLE POINTS IN THE SIDELobe REGION. *
28 C    * NS - NUMBER OF RESTARTS. *
29 C    * NI - NUMBER OF CONJUGATE GRADIENT ITERATIONS PER RESTART. *
30 C    *
31 C    * OUTPUTS:
32 C    *
33 C    * U(NE) - REAL PART OF THE EXCITATION OF EACH ELEMENT. *
34 C    * V(NE) - IMAGINARY PART OF THE EXCITATION OF EACH ELEMENT. *
35 C    *****
36 C    PARAMETER (NL=6,NTE=43,NBE=0,NE=43,NSP=700)
37 C    REAL*4 U(NE),X(NE),Y(NE),S(NE),GN(NE),H(NE,NF),P(NE),Q(NE),R(NE)
38 C    REAL*4 Z(NE),KI(NSP),KY(NSP),KXP,KYP,NWBP
39 C    INTEGER IP(NSP),BAD(NE)
40 C    PI=.3141593E+01
41 C    OPEN(UNIT=2,FILE='OUTPUT',STATUS='UNKNOWN')
42 C    OPEN(UNIT=4,FILE='ESTIMATE',STATUS='UNKNOWN')
43 C    *****
44 C    * GENERATE THE GEOMETRY. *
45 C    *****
46 C    DS=0.6
47 C    BAD(1)=15
48 C    BAD(2)=17
49 C    BAD(3)=20

```



```

50      CALL GEOMET(NL,DS,NE,NBE,BAD,X,Y)
51 C    *****
52 C    * INITIALIZE THE TAPER DISTRIBUTION. *
53 C    *****
54      IFLAG1=1
55      SLL=-40.0
56      NBAR=10
57      IF (IFLAG1.EQ. 1) THEN
58          CALL TAPER(NE,U,X,Y,SLL,NBAR,NL,DS)
59          REWIND 4
60          WRITE(4,101) (I,U(I),I=1,NE)
61      ELSE
62          END IF
63          REWIND 4
64          READ(4,101) (I,U(I),I=1,NE)
65 C    *****
66 C    * COMPUTE THE GAIN (DIRECTIVITY) OF THE ARRAY. *
67 C    *****
68 C    CALL GAIN(NE,U,X,Y,GDB)
69 C    *****
70 C    * FIND THE HALF POWER NULL WIDTH BETWEEN PEAKS. *
71 C    *****
72 C    CALL BEAM(NE,U,X,Y,NWBP)
73 C    *****
74 C    * GENERATE THE K-SPACE SAMPLE POINTS. *
75 C    *****
76      BBF=1.24
77      CALL KSPACE(NMP,NL,DS,SLL,NBAR,BBF,KX,KY)
78 C    *****
79 C    * COMPUTE NORMALIZED PEAK SIDELobe POWER IN DB. *
80 C    *****
81      CALL PEKPOW(NE,U,X,Y,KIP,KYP,PO)
82      NP=0
83      CALL PEAKSL(NE,U,X,Y,NSP,NMP,KX,KY,IP,NP,PMAX)
84      PMAXDB=10.0*ALOG10(PMAX/PO)
85 C    *****
86 C    * PERFORM CONJUGATE GRADIENT STEPS. *
87 C    *****
88      NS=150
89      NI=3
90      CALL CGRAD(NE,NSP,NMP,NS,NI,U,X,Y,S,NP,IP,KX,KY,GN,H,P,Q,R,Z)
91      REWIND 4
92      WRITE(4,101) (I,U(I),I=1,NE)
93 C    *****
94 C    * POWER PATTERN GRAPHICS OUTPUT. *
95 C    *****
96      REWIND 4
97      READ(4,101) (I,U(I),I=1,NE)
98 C    *****

```

```

99 C      * PRINT THE POWER PATTERN IN CYLINDRICAL COORDINATES.      *
100 C      *****
101      CALL POWPAT(NE,U,X,Y)
102 C      *****
103 C      * PRINT THE POWER PATTERN IN SPHERICAL COORDINATES.      *
104 C      *****
105      CALL SPOWPAT(NE,U,X,Y)
106 C      *****
107 C      * FORMATS.      *
108 C      *****
109 100    FORMAT(I5,F15.5,1X,F15.5)
110 101    FORMAT(I5,F15.7)
111      END
112 C
113      SUBROUTINE GEOMET(NL,S,NE,NBE,BAD,X,Y)
114 C      *****
115 C      * COMPUTE THE POSITION OF EACH ELEMENT IN A HEXOGONAL ARRAY.      *
116 C      *****
117      REAL*4 X(NE),Y(NE)
118      INTEGER BAD(NE),IFLAG
119      HGT=(SQRT(3.0)/2.0)*S
120 C      *****
121 C      * CENTER ROW.      *
122 C      *****
123      K=0
124      L=0
125      XMIN=-NL*S
126      DO 1 I=0,2*NL
127          XX=XMIN+I*S
128          IF (XX.GT. 0.001) THEN
129              L=L+1
130              CALL CHECK(L,NE,NBE,BAD,IFLAG)
131              IF (IFLAG.EQ. 0) THEN
132                  K=K+1
133                  X(K)=XX
134                  Y(K)=0.0
135              ELSE
136                  END IF
137              ELSE
138                  END IF
139 1      CONTINUE
140 C      *****
141 C      * TOP ROWS.      *
142 C      *****
143      INUM=2*NL+1
144      DO 2 J=1,NL
145          INUM=INUM-1
146          XMIN=-NL*S+J*S/2.0
147          DO 3 I=0,INUM-1

```

```

148      II=IMIN+I*S
149      IF (II .GT. 0.001) THEN
150          L=L+1
151          CALL CHECK(L,NE,NBE,BAD,IFLAG)
152          IF (IFLAG .EQ. 0) THEN
153              K=K+1
154              X(K)=II
155              Y(K)=HGT*J
156          ELSE
157              END IF
158          ELSE
159              END IF
160      3      CONTINUE
161      2      CONTINUE
162  C *****
163  C * BOTTOM ROWS.
164  C *****
165      INUM=2-NL+1
166      DO 4 J=1,NL
167          INUM=INUM-1
168          IMIN=-NL*S+J*S/2.0
169          DO 5 I=0,INUM-1
170              IX=IMIN+I*S
171              IF (IX .GT. 0.001) THEN
172                  L=L+1
173                  CALL CHECK(L,NE,NBE,BAD,IFLAG)
174                  IF (IFLAG .EQ. 0) THEN
175                      K=K+1
176                      X(K)=IX
177                      Y(K)=-HGT*J
178                  ELSE
179                      END IF
180                  ELSE
181                      END IF
182      5      CONTINUE
183      4      CONTINUE
184      RETURN
185      END
186  C
187      SUBROUTINE CHECK(L,NE,NBE,BAD,IFLAG)
188  C *****
189  C * CHECK TO SEE IF THAT ELEMENT IS BAD.
190  C *****
191      INTEGER BAD(NE)
192      IFLAG=0
193      DO 1 I=1,NBE
194          IF (L .EQ. BAD(I)) THEN
195              IFLAG=1
196          ELSE

```

```

107         END IF
108 1        CONTINUE
109         RETURN
200         END
201 C
202         SUBROUTINE BEAM(NE,U,X,Y,BWFN)
203 C        *****
204 C        * COMPUTE THE BEAM WIDTH BETWEEN FIRST NULLS. *
205 C        *****
206         REAL U(NE),X(NE),Y(NE),KXI,KYI,KIP,KYP
207         RAD=.17453293E-01
208         TP=.0283185E+01
209         TMIN=0.0
210         TMAX=12.0
211         P=0.0
212         NT=140
213         DT=(TMAX-TMIN)/NT
214 C        *****
215 C        * FIND THE MAXIMUM VALUE. *
216 C        *****
217         CALL PEKPOW(NE,U,X,Y,KIP,KYP,PEAK)
218 C        *****
219 C        * RECOMPUTE AND NOW FIND THE POINT WHICH IS AT THE SLL BELOW THE *
220 C        * PEAK. *
221 C        *****
222         DO 3 I=0,NT
223             T=TMIN+I*DT
224             TPS=TP*SIN(RAD*T)
225             KXI=TPS
226             CALL POWI(NE,U,X,Y,KXI,0.0,POW)
227             IF (POW/PEAK .GE. 0.5) THEN
228                 WRITE(2,100) T,10.0*ALOG(POW/PEAK)
229                 GO TO 99
230             ELSE
231                 END IF
232 3        CONTINUE
233 100      FORMAT(F10.5,1X,F10.5)
234 99      CONTINUE
235         BWFN=2.0*T
236         RETURN
237         END
238 C
239         SUBROUTINE KSPACE(NMP,NL,DS,SLL,NBAR,BBF,KI,KY)
240 C        *****
241 C        * THIS SUBROUTINE SAMPLES THE KI >0 REGION OF K SPACE. *
242 C        *****
243         REAL*4 KX(*),KY(*),KC,KR,KXI,KYY,KIMIN,KIMAX,KYMIN,KYMAX,KNULL
244         RAD=.17453293E-01
245         PI=.3141593E+01

```

```

246      TP=.6283186E+01
247 C      *****
248 C      * COMPUTE THE FIRST ZERO OF THE THEORETICAL BAYLISS PATTERN.      *
249 C      *****
250      S=ABS(SLL)
251      C1=3.038753E-01
252      C2=5.042922E-02
253      C3=-2.7989E-04
254      C4=3.43E-06
255      C5=-2.0E-8
256      A=C1+S*(C2+S*(C3+S*(C4+S*C5)))
257      C1=9.858302E-01
258      C2=3.33885E-02
259      C3=1.4064E-04
260      C4=-1.9E-06
261      C5=1.0E-08
262      ZETA1=C1+S*(C2+S*(C3+S*(C4+S*C5)))
263      U1=(NBAR+0.6)*SQRT(ZETA1+ZETA1/(A+A+NBAR))
264 C      *****
265 C      * COMPUTE THE K VALUE OF THE FIRST NULL.      *
266 C      *****
267      R=NL*DS
268      KNULL=U1/R
269 C      *****
270 C      * COMPUTE THE MAXIMUM K VALUE ALLOWED FOR THE FIRST NULL.      *
271 C      *****
272 C      KC=TP*0.46
273      KC=BBF*KNULL
274 C      *****
275 C      * GENERATE THE RECTANGULAR LATTICE OF SAMPLE POINTS.      *
276 C      *****
277      KYMIN=-TP
278      KYMAX=TP
279      KIMIN=0.0
280      KIMAX=TP
281      NKX=27
282      NKY=46
283      DKX=(KXMAX-KIMIN)/(NKX-1)
284      DKY=(KYMAX-KYMIN)/(NKY-1)
285      NMP=0
286      DO 1 I=0,NKY
287          KYY=KYMIN+I*DKY
288          DO 2 J=0,NKX
289              KXX=KIMIN+J*DKX
290              KR=SQRT(KIX*KXX+KYY*KYY)
291              IF ((KR .GT. KC) .AND. (KR .LT. TP)
292                  * .AND. (KIX .GT. 0.0)) THEN
293                  NMP=NMP+1
294                  KX(NMP)=KIX

```

```

295         KY(NMP)=KYY
296     ELSE
297     END IF
298 2     CONTINUE
299 1     CONTINUE
300     RETURN
301     END
302 C
303     SUBROUTINE TAPER(NE,U,X,Y,SLL,NBAR,NL,DS)
304 C *****
305 C * INITIALIZE THE AMPLITUDE DISTRIBUTION. *
306 C * NOTE THAT THE AMPLITUDE=U*U. THE BAYLISS DISTRIBUTION FORMULAS *
307 C * ARE TAKEN FROM MODERN ANTENNA DESIGN BY T. MILLIGAN. *
308 C *****
309     REAL*4 U(NE),X(NE),Y(NE),ROOT(20),ZETA(4),UZER(20),B(20)
310     PI=.3141593E+01
311 C *****
312 C * DEFINE THE ROOTS. *
313 C *****
314     ROOT(1)=0.5860670
315     ROOT(2)=1.6970609
316     ROOT(3)=2.7171939
317     ROOT(4)=3.7261370
318     ROOT(5)=4.7312271
319     ROOT(6)=5.7345206
320     ROOT(7)=6.7368221
321     ROOT(8)=7.7385356
322     ROOT(9)=8.7398506
323     ROOT(10)=9.740896
324     ROOT(11)=10.7417435
325     ROOT(12)=11.7424476
326     ROOT(13)=12.7430408
327     ROOT(14)=13.7435477
328     ROOT(15)=14.7439866
329     ROOT(16)=15.7443679
330     ROOT(17)=16.7447044
331     ROOT(18)=17.7450030
332     ROOT(19)=18.7452697
333     ROOT(20)=19.7455093
334 C *****
335 C * COMPUTE APPROXIMATE ARRAY RADIUS. *
336 C *****
337     RADIUS=DS*NL
338 C *****
339 C * CHOOSE SIDELOBE LEVEL. *
340 C *****
341     CALL COEFF(SLL,A,ZETA)
342     CALL NEWZ(A,ZETA,NBAR,ROOT,UZER)
343     CALL FBCOF(ROOT,UZER,NBAR,B)

```

```

344 C *****
345 C * COMPUTE AMPLITUDE. *
346 C *****
347 DO 1 K=1,NE
348 R=SQRT(I(K)*I(K)+Y(K)*Y(K))
349 COSPHI=I(K)/R
350 SUM=0.0
351 DO 2 M=1,NBAR
352 RHO=R/RADIUS
353 ARG=PI*ROOT(M)*RHO
354 CALL J1(ARG,BESS1)
355 SUM=SUM+B(M)*BESS1
356 2 CONTINUE
357 AMP=COSPHI*SUM
358 U(K)=AMP
359 1 CONTINUE
360 CALL NORMAL(NE,U)
361 DO 3 I=1,NE
362 WRITE(2,*) I,X(I),Y(I),R,RADIUS,U(I)
363 3 CONTINUE
364 100 FORMAT(I5,1X,F10.4,1X,F10.4,1X,F10.4)
365 RETURN
366 END
367 C
368 SUBROUTINE COEFF(SLL,A,ZETA)
369 REAL*4 ZETA(4)
370 S=ABS(SLL)
371 C1=3.038763E-01
372 C2=5.042922E-02
373 C3=-2.7989E-04
374 C4=3.43E-06
375 C5=-2.0E-8
376 A=C1+S*(C2+S*(C3+S*(C4+S*C5)))
377 C1=9.858302E-01
378 C2=3.33865E-02
379 C3=1.4064E-04
380 C4=-1.9E-06
381 C5=1.0E-08
382 ZETA(1)=C1+S*(C2+S*(C3+S*(C4+S*C5)))
383 C1=2.00337487
384 C2=1.141548E-02
385 C3=4.169E-04
386 C4=-3.73E-06
387 C5=1.0E-08
388 ZETA(2)=C1+S*(C2+S*(C3+S*(C4+S*C5)))
389 C1=3.00636321
390 C2=6.83394E-03
391 C3=2.9281E-04
392 C4=-1.61E-06

```

```

393      ZETA(3)=C1+S*(C2+S*(C3+S*C4))
394      C1=4.00518432
395      C2=5.01795E-03
396      C3=2.1735E-04
397      C4=-8.8E-07
398      ZETA(4)=C1+S*(C2+S*(C3+S*C4))
399      RETURN
400      END
401 C
402      SUBROUTINE NEWZ(A,ZETA,NBAR,ROOT,UZER)
403      REAL*4 ZETA(4),ROOT(20),UZER(20)
404      DO 1 N=1,4
405          ARG=ZETA(N)*ZETA(N)/(A*A+NBAR*NBAR)
406          UZER(N)=ROOT(NBAR+1)*SQRT(ARG)
407 1      CONTINUE
408      DO 2 N=5,NBAR-1
409          ARG=(A*A+N*N)/(A*A+NBAR*NBAR)
410          UZER(N)=ROOT(NBAR+1)*SQRT(ARG)
411 2      CONTINUE
412      RETURN
413      END
414 C
415      SUBROUTINE FBCOF(ROOT,UZER,NBAR,B)
416      REAL*4 ROOT(20),UZER(20),B(20)
417      REAL*8 RAT,UN2,RM2,RN2,PR1,PR2
418      PI=3.141593
419      DO 1 M=1,NBAR
420          RM2=ROOT(M)*RCOT(M)
421          PR1=1.0
422          DO 2 N=1,NBAR-1
423              UN2=UZER(N)*UZER(N)
424              RAT=1.0-RM2/UN2
425              PR1=PR1*RAT
426 2      CONTINUE
427          PR2=1.0
428          DO 3 N=1,NBAR
429              IF (N.NE.M) THEN
430                  RN2=ROOT(N)*ROOT(N)
431                  RAT=1.0-RM2/RN2
432                  PR2=PR2*RAT
433              ELSE
434                  END IF
435 3      CONTINUE
436          ARG=PI*ROOT(M)
437          CALL J1(ARG,BESS1)
438          B(M)=(RM2/BESS1)*(PR1/PR2)
439 1      CONTINUE
440      RETURN
441      END

```



```

442 C
443     SUBROUTINE J1(X,BESS1)
444 C*****C
445 C THIS SUBROUTINE COMPUTES J1(X) C
446 C*****C
447     A0=-0.50249985
448     A1=0.21093573
449     A2=-0.03954289
450     A3=0.00443319
451     A4=-0.00031761
452     A5=0.00001109
453     B0=0.79788456
454     B1=0.00000166
455     B2=0.01659667
456     B3=0.00017105
457     B4=-0.00249511
458     B5=0.00113653
459     B6=-0.00020033
460     C0=-2.35619449
461     C1=0.12499612
462     C2=0.00006660
463     C3=-0.00037879
464     C4=0.00074348
465     C5=0.00079824
466     C6=-0.00029166
467     IF ((X .GE. 0.0) .AND. (X .LE. 3.0)) THEN
468         S=(X/3.0)*(X/3.0)
469         P1=S*(A0+S*(A1+S*(A2+S*(A3+S*(A4+A5*S))))
470         BESS1=X*(0.5+P1)
471     ELSE IF (X .GT. 3.0) THEN
472         T=3.0/X
473         Q1=B0+T*(B1+T*(B2+T*(B3+T*(B4+T*(B5+B6*T))))
474         V1=X+C0+T*(C1+T*(C2+T*(C3+T*(C4+T*(C5+C6*T))))
475         BESS1=Q1*COS(V1)/SQRT(X)
476     ELSE
477         END IF
478     RETURN
479     END
480 C
481     SUBROUTINE CGRAD(NE,NSP,NMP,NS,NI,U,X,Y,S,NP,IP,KX,KY,GN,H,P,Q,R
482     &,Z)
483 C *****
484 C * COMPUTE POWER IN A SINGLE DIRECTION. *
485 C *****
486     REAL*4 U(NE),X(NE),Y(NE),S(NE),GN(NE),H(NE,NE),P(NE),Q(NE),R(NE)
487     REAL*4 Z(NE),KX(NSP),KY(NSP),KXI,KYI,KIP,KYP
488     INTEGER IP(NSP)
489 C *****
490 C * INITIALIZE THE NUMBER OF SIDELobe SAMPLE POINTS. *

```

```

491 C *****
492 C      NP=0
493 C *****
494 C * PERFORM NS RE-STARTS. *
495 C *****
496 C      DO 1 L=1,NS
497 C          *****
498 C          * FIND THE PEAK SIDELobe POWER POINT AND UPDATE THE NUMBER *
499 C          * OF MATCH POINTS. *
500 C          *****
501 C          CALL PEAKSL(NE,U,X,Y,NSP,NMP,KX,KY,IP,NP,PMAI)
502 C          *****
503 C          * FIND THE POWER IN THE MAIN BEAM. *
504 C          *****
505 C          CALL PEKPOW(NE,U,X,Y,KXP,KYP,PEAK)
506 C          PO=PEAK
507 C          *****
508 C          * COMPUTE AND STOKE AN ARRAY OF THE 1ST DERIVATIVE OF THE *
509 C          * POWER IN THE MAIN BEAM DIRECTION WITH RESPECT TO THE MTH *
510 C          * VARIABLE. *
511 C          *****
512 C          DO 2 M=1,NE
513 C              CALL DPOWI(NE,U,X,Y,M,KXP,KYP,DPWI)
514 C              S(M)=DPWI
515 C          CONTINUE
516 C          *****
517 C          * COMPUTE AND STORE THE AVERAGE SIDELobe POWER. *
518 C          *****
519 C          PA=0.0
520 C          DO 3 I=1,NP
521 C              CALL POWI(NE,U,X,Y,KX(IP(I)),KY(IP(I)),POWERI)
522 C              PA=PA+POWERI
523 C          CONTINUE
524 C          PA=PA/NP
525 C          WRITE(*,200) L,10.0*ALOG10(PA/PO),10.0*ALOG10(PMAI/PO)
526 C          WRITE(2,200) L,10.0*ALOG10(PA/PO),10.0*ALOG10(PMAI/PO)
527 200  FORMAT(1X,I5,1X,'AVE PEAK SLL',1X,F15.6,1X,
528 C          &          'PEAK SLL',1X,F15.6)
529 C          *****
530 C          * COMPUTE THE NEGATIVE OF THE GRADIENT. *
531 C          *****
532 C          DO 4 M=1,NE
533 C              *****
534 C              * COMPUTE THE DERIVATIVE OF THE AVERAGE SIDELobe POWER. *
535 C              *****
536 C              DPAM=0.0
537 C              DO 5 I=1,NP
538 C                  CALL DPOWI(NE,U,X,Y,M,KX(IP(I)),KY(IP(I)),DPWI)
539 C                  DPAM=DPAM+DPWI

```

```

540 B          CONTINUE
541          DPAM=DPAM/NP
542 C          *****
543 C          * COMPUTE THE NEGATIVE GRADIENT.          *
544 C          *****
545          GN(M)=(PA*S(M)/PO-DPAM)/PO
546 4          CONTINUE
547 C          *****
548 C          * COMPUTE THE HESSIAN.          *
549 C          *****
550          DO 6 M=1,NE
551              DO 7 N=M,NE
552                  CALL DDPOWI(NE,U,X,Y,M,N,KXP,KYP,DDPWI)
553                  DDPO=DDPWI
554                  DDPA=0.0
555                  DO 8 I=1,NP
556                      CALL DDPOWI(NE,U,X,Y,M,N,KI(IP(I)),KY(IP(I)),DDPWI)
557                      DDPA=DDPA+DDPWI
558 8          CONTINUE
559          DDPA=DDPA/NP
560          H(M,N)=(GN(N)*S(M)+GN(M)*S(N)+DDPA-PA*DDPO/PO)/PO
561          H(N,M)=H(M,N)
562 7          CONTINUE
563 6          CONTINUE
564 C          *****
565 C          * START THE CONJUGATE GRADIENT ALGORITHM.          *
566 C          *****
567 C          *****
568 C          * INITIALIZE THE RESIDUAL.          *
569 C          *****
570          DO 12 M=1,NE
571              R(M)=GN(M)
572              Z(M)=0.0
573 12          CONTINUE
574 C          *****
575 C          * INITIALIZE SEARCH VECTOR.          *
576 C          *****
577          CALL MATVEC(NE,H,R,Q)
578          CALL NORM22(NE,Q,QN)
579          BE0=1.0/QN
580          DO 13 I=1,NE
581              P(I)=BE0*Q(I)
582 13          CONTINUE
583 C          *****
584 C          * PERFORM CONJUGATE GRADIENT ITERATIONS.          *
585 C          *****
586          DO 14 K=1,NI
587 C          *****
588 C          * UPDATE AMPLITUDE VECTOR AND RESIDUAL.          *

```

```

589 C *****
590 CALL MATVEC(NE,H,P,Q)
591 CALL NORM22(NE,Q,QN)
592 AK=1.0/QN
593 DO 15 I=1,NE
594     Z(I)=Z(I)+AK*P(I)
595     R(I)=R(I)-AK*Q(I)
596 15 CONTINUE
597 CALL NORM22(NE,R,RN)
598 ERR=SQRT(RN)
599 WRITE(*,500) L,K,ERR
600 WRITE(2,500) L,K,ERR
601 500 FORMAT(3X,'RESIDUAL',1X,I3,1X,I3,1X,E15.8)
602 C *****
603 C * UPDATE SEARCH VECTOR. *
604 C *****
605 CALL MATVEC(NE,H,R,Q)
606 CALL NORM22(NE,Q,QN)
607 BEK=1.0/QN
608 DO 16 I=1,NE
609     P(I)=P(I)+BEK*Q(I)
610 16 CONTINUE
611 14 CONTINUE
612 C *****
613 C * UPDATE AMPLITUDE VECTOR. *
614 C *****
615 DO 17 I=1,NE
616     U(I)=U(I)+Z(I)
617 17 CONTINUE
618 CALL NORMAL(NE,U)
619 C *****
620 C * WRITE OUT THE NEW ESTIMATE TO A FILE. *
621 C *****
622 REWIND 4
623 WRITE(4,101) (I,U(I),I=1,NE)
624 101 FORMAT(I5,F15.7)
625 IF (NP .EQ. NMP) THEN
626     GO TO 99
627 ELSE
628     END IF
629 1 CONTINUE
630 99 CONTINUE
631 RETURN
632 END
633 C
634 SUBROUTINE PEAKSL(NE,U,X,Y,NSP,NMP,KX,KY,IP,NP,PMAX)
635 C *****
636 C * THIS SUBROUTINE FINDS THE PEAK SIDELobe POWER LEVEL COORDINATE *
637 C * POINT (KX(LMAX),KY(LMAX)). IF THE POINT IS NEW THEN IT IS *

```

```

038 C      * ACCUMULATED INTO THE POINTER ARRAY IP. THE NUMBER OF POINTS IN *
039 C      * THE INTEGER POINTER ARRAY IP IS NP SUCH THAT IP(I),I=1...NP. *
040 C      *****
041      REAL U(NE),X(NE),Y(NE) KX(NSP),KY(NSP)
042      INTEGER IP(NSP)
043 C      *****
044 C      * COMPUTE THE POWER AT EACH POINT AND FIND THE MAX. *
045 C      *****
046      CALL POWI(NE,U,X,Y,KX(1),KY(1),PWI)
047      PMAX=PWI
048      LMAX=1
049      DO 1 I=1,NMP
050          CALL POWI(NE,U,X,Y,KX(I),KY(I),PWI)
051          IF (PWI .GT. PMAX) THEN
052              LMAX=I
053              PMAX=PWI
054          ELSE
055              END IF
056 1      CONTINUE
057 C      *****
058 C      * COMPARE LMAX TO ALL THE PREVIOUS POINTERS IP(I),I=1...NP. *
059 C      * IF LMAX MATCHES ONE OF THE PREVIOUS POINTERS THEN THE *
060 C      * POINTER ARRAY NEED NOT BE UPDATED (IFLAG=1). IF LMAX DOES *
061 C      * NOT MATCH A PREVIOUS POINTER VALUE (IFLAG=0) THEN IT *
062 C      * BECOMES THE NP+1TH ELEMENT OF THE POINTER ARRAY. *
063 C      *****
064      IFLAG=0
065      DO 2 I=1,NP
066          IF (IP(I) .EQ. LMAX) THEN
067              IFLAG=1
068          ELSE
069              END IF
070 2      CONTINUE
071      IF (IFLAG .EQ. 0) THEN
072          NP=NP+1
073          IP(NP)=LMAX
074      ELSE
075          END IF
076      RETURN
077      END
078 C
079      SUBROUTINE POWI(NE,U,X,Y,KXI,KYI,POWERI)
080 C      *****
081 C      * COMPUTE POWER IN A SINGLE DIRECTION. *
082 C      *****
083      REAL U(NE),X(NE),Y(NE),KXI,KYI
084      SUM=0.0
085      DO 1 K=1,NE
086          PSI=KXI*(X(K)+KYI*Y(K)

```

```

687      SUM=SUM+U(K)*SIN(PSI)
688 1     CONTINUE
689      POWERI=SUM*SUM
690      RETURN
691      END

692 C
693      SUBROUTINE DPOWI(NE,U,X,Y,M,KXI,KYI,DPWI)
694 C      *****
695 C      * COMPUTE 1ST DERIVATIVE OF POWER WITH RESPECT TO THE MTH VARIABLE*
696 C      * IN THE ITH DIRECTION. *
697 C      *****
698      REAL U(NE),X(NE),Y(NE),KXI,KYI
699 C      *****
700 C      * COMPUTE TI AND VI. *
701 C      *****
702      SUM=0.0
703      DO 1 K=1,NE
704          PSI=KXI*X(K)+KYI*Y(K)
705          SUM=SUM+U(K)*SIN(PSI)
706 1     CONTINUE
707 C      *****
708 C      * COMPUTE 1ST DERIVATIVE OF TI AND VI. *
709 C      *****
710      PSIM=KXI*X(M)+KYI*Y(M)
711      DPWI=2.0*SIN(PSIM)*SUM
712      RETURN
713      END

714 C
715      SUBROUTINE DDPOWI(NE,U,X,Y,M,N,KXI,KYI,DDPWI)
716 C      *****
717 C      * COMPUTE 2ND DERIVATIVE OF POWER WITH RESPECT TO THE M AND NTH *
718 C      * VARIABLES IN THE ITH DIRECTION. *
719 C      *****
720      REAL*4 U(NE),X(NE),Y(NE),KXI,KYI
721 C      *****
722 C      * COMPUTE 2ND DERIVATIVE OF TI AND VI WITH RESPECT TO M,N. *
723 C      *****
724      PSIM=KXI*X(M)+KYI*Y(M)
725      PSIN=KXI*X(N)+KYI*Y(N)
726 C      *****
727 C      * COMPUTE 2ND DERIVATIVE OF POWER. *
728 C      *****
729      DDPWI=2.0*SIN(PSIM)*SIN(PSIN)
730      RETURN
731      END

732 C
733      SUBROUTINE NORM22(N,A,AN)
734 C      *****
735 C      * THIS SUBROUTINE COMPUTES THE EUCLIDIAN NORM SQUARED OF A. *

```

```

736 C *****
737 REAL*4 A(*)
738 AN=0.0
739 DO 1 I=1,N
740     AN=AN+A(I)*A(I)
741 1 CONTINUE
742 RETURN
743 END
744 C
745 SUBROUTINE MATVEC(N,H,P,Q)
746 C *****
747 C * THIS SUBROUTINE COMPUTES THE MATRIX PRODUCT Q=HP. *
748 C *****
749 REAL*4 H(N,N),P(N),Q(N)
750 DO 1 I=1,N
751     SUM=0.0
752     DO 2 J=1,N
753         SUM=SUM+H(I,J)*P(J)
754 2 CONTINUE
755     Q(I)=SUM
756 1 CONTINUE
757 RETURN
758 END
759 C
760 SUBROUTINE NORMAL(N,A)
761 C *****
762 C * THIS SUBROUTINE NORMALIZES A VECTOR OF LENGTH N. *
763 C *****
764 REAL*4 A(N)
765 VMAX=A(1)
766 DO 1 I=1,N
767     IF (A(I) .GT. VMAX) THEN
768         VMAX=A(I)
769     ELSE
770         END IF
771 1 CONTINUE
772 DO 2 I=1,N
773     A(I)=A(I)/ABS(VMAX)
774 2 CONTINUE
775 RETURN
776 END
777 C
778
779 SUBROUTINE PEKPOW(NE,U,X,Y,KXP,KYP,PEAK)
780 C *****
781 C * COMPUTE POWER IN A SINGLE DIRECTION GIVEN A THETA AND PHI. *
782 C *****
783 REAL U(NE),X(NE),Y(NE),KXI,KYI,KXP,KYP
784 RAD=.17453293E-01

```

```

785      TP=.6283185E+01
786      TMIN=0.0
787      TMAX=30.0
788      PMIN=-10.0
789      PMAX=10.0
790      NT=66
791      NP=46
792      DT=(TMAX-TMIN)/NT
793      DP=(PMAX-PMIN)/NP
794      DO 1 I=0,NT
795          T=TMIN+I*DT
796          TPS=TP*SIN(RAD*T)
797          DO 2 J=0,NP
798              P=PMIN+J*DP
799              RP=RAD*P
800              KXI=TPS*COS(RP)
801              KYI=TPS*SIN(RP)
802              CALL POWI(NE,U,X,Y,KXI,KYI,POW)
803              IF (I.EQ. 1) THEN
804                  PEAK=POW
805                  KXP=KXI
806                  KYP=KYI
807                  THEPK=T
808                  PHIPK=P
809              ELSE IF (POW.GT. PEAK) THEN
810                  PEAK=POW
811                  KXP=KXI
812                  KYP=KYI
813                  THEPK=T
814                  PHIPK=P
815              ELSE
816                  END IF
817      2      CONTINUE
818      1      CONTINUE
819      WRITE(*,*) THEPK,PHIPK,KXP,KYP,PEAK
820      WRITE(2,*) THEPK,PHIPK,KXP,KYP,PEAK
821      RETURN
822      END
823  C
824      SUBROUTINE GAIN(NE,U,X,Y,GDB)
825  C      *****
826  C      * COMPUTE POWER IN A SINGLE DIRECTION. *
827  C      *****
828      REAL U(NE),X(NE),Y(NE),KIP,KYP
829      RAD=.17453293E-01
830      PI=.3141593E+01
831      TP=.6283185E+01
832  C      *****
833  C      * GRAPH INPUTS

```



```

834 C *****
835     NP=41
836     NR=36
837     PMIN=0.0
838     PMAX=360.0
839     TMIN=0.0
840     TMAX=90.0
841     DP=(PMAX-PMIN)/(NP-1)
842     DR=(TMAX-TMIN)/(NR-1)
843 C *****
844 C     * COMPUTE PEAK VALUE. *
845 C *****
846     CALL PEKPOW(NE,U,X,Y,KIP,KYP,PEAK)
847 C *****
848 C     * COMPUTE THE SOLID ANGLE. *
849 C *****
850     SUM=0.0
851     DO 1 I=0,NR-2
852         T=TMIN+DR/2.0+I*DR
853         ST=SIN(RAD*T)
854         DO 2 J=0,NP-2
855             P=PMIN+DP/2.0+J*DP
856             CALL POWER(NE,U,X,Y,T,P,POW)
857             SUM=SUM+ST*POW
858 2     CONTINUE
859 1     CONTINUE
860     SOLID=RAD*RAD*DR*DP*SUM/PEAK
861     GDB=10.0*ALOG10(TP/SOLID)
862     RETURN
863     END
864 C
865     SUBROUTINE POWPAT(NE,U,X,Y)
866 C *****
867 C     * COMPUTE POWER IN A SINGLE DIRECTION. *
868 C *****
869     REAL U(NE),X(NE),Y(NE),KIP,KYP
870     OPEN(UNIT=7,FILE='DATA')
871     RAD=.17453293E+01
872     PI=.3141593E+01
873     TP=.6283185E+01
874 C *****
875 C     * GRAPH INPUTS *
876 C *****
877     NP=41
878     NR=25
879     NS=0
880     PMIN=0.0
881     PMAX=360.0
882     TMIN=0.0

```

```

883      TMAX=90.0
884      DP=(PMAX-PMIN)/(NP-1)
885      DR=(TMAX-TMIN)/(NR-1)
886      AX=150.0
887      AY=AX
888      AZ=680.0
889      PO=65.0
890      QO=60.0
891      XA=-2.0
892      XB=2.0
893      YA=-2.0
894      YB=2.0
895      ZA=-2.0
896      ZB=2.3
897      AT=.3
898      WRITE(7,*) NP,NR,NS,AX,AY,AZ
899      WRITE(7,*) XA,XB,YA,YB,ZA,ZB
900      WRITE(7,*) AT,PO,QO
901 C      *****
902 C      * PHI VALUES. *
903 C      *****
904      DO 1 I=0,NP-1
905          P=PMIN+I*DP
906          WRITE(7,*) RAD*P
907 1      CONTINUE
908 C      *****
909 C      * RADIAL (THETA) VALUES. *
910 C      *****
911      DO 2 I=0,NR-1
912          T=TMIN+I*DR
913          WRITE(7,*) RAD*T
914 2      CONTINUE
915 C      *****
916 C      * FUNCTION VALUES. *
917 C      *****
918 C      *****
919 C      * COMPUTE PEAK VALUE. *
920 C      *****
921      CALL PEKPOW(NE,U,X,Y,KIP,KYP,PEAK)
922 C      *****
923 C      * WRITE OUT THE NORMALIZED POWER. *
924 C      *****
925      DO 5 I=0,NR-1
926          T=TMIN+I*DR
927          DO 6 J=0,NP-1
928              P=PMIN+J*DP
929              CALL POWER(NE,U,X,Y,T,P,POW)
930              WRITE(7,*) POW/PEAK
931 6      CONTINUE

```

```

032 5      CONTINUE
033      RETURN
034      END
035 C
036      SUBROUTINE SPOWPAT(NE,U,I,Y)
037 C      *****
038 C      * COMPUTE POWER IN A SINGLE DIRECTION. *
039 C      *****
040      REAL U(NE),I(NE),Y(NE)
041      OPEN(UNIT=8,FILE='SDATA')
042      RAD=.17453293E-01
043      PI=.3141593E+01
044      TP=.6283185E+01
045 C      *****
046 C      * GRAPH INPUTS *
047 C      *****
048      NP=119
049      NR=53
050      NS=0
051      PMIN=0.0
052      PMAX=360.0
053      TMIN=0.0
054      TMAX=90.0
055      DP=(PMAX-PMIN)/(NP-1)
056      DR=(TMAX-TMIN)/(NR-1)
057      AX=600.0
058      AY=AX
059      AZ=AX
060      PO=65.0
061      QO=60.0
062      XA=-2.0
063      XB=2.0
064      YA=-2.0
065      YB=2.0
066      ZA=-2.0
067      ZB=2.3
068      AT=.3
069      IT=4
070      WRITE(8,*) NP,NR,NS,II,AX,AY,AZ
071      WRITE(8,*) XA,XB,YA,YB,ZA,ZB
072      WRITE(8,*) AT,PO,QO
073 C      *****
074 C      * PHI VALUES. *
075 C      *****
076      DO 1 I=0,NP-1
077          P=PMIN+I*DP
078          WRITE(8,*) RAD*P
079 1      CONTINUE
080 C      *****

```

```

981 C      * RADIAL (THETA) VALUES.
982 C      *****
983      DO 2 I=0,NR-1
984          T=TMIN+I*DR
985          WRITE(8,*) RAD*T
986 2      CONTINUE
987 C      *****
988 C      * FUNCTION VALUES.
989 C      *****
990 C      *****
991 C      * COMPUTE PEAK VALUE.
992 C      *****
993          PEAK=-100.0
994      DO 3 I=0,NR-1
995          T=TMIN+I*DR
996          DO 4 J=0,NP-1
997              P=PMIN+J*DP
998              CALL POWER(NE,U,I,Y,T,P,POW)
999              IF (POW .GT. PEAK) THEN
1000                  PEAK=POW
1001              ELSE
1002                  END IF
1003 4          CONTINUE
1004 3      CONTINUE
1005 C      *****
1006 C      * WRITE OUT THE NORMALIZED POWER.
1007 C      *****
1008      DO 5 I=0,NR-1
1009          T=TMIN+I*DR
1010          DO 6 J=0,NP-1
1011              P=PMIN+J*DP
1012              CALL POWER(NE,U,I,Y,T,P,POW)
1013              WRITE(8,*) POW/PEAK
1014 6          CONTINUE
1015 5      CONTINUE
1016      RETURN
1017      END
1018 C
1019      SUBROUTINE POWER(NE,U,I,Y,T,P,POW)
1020 C      *****
1021 C      * COMPUTE POWER IN A SINGLE DIRECTION GIVEN A THETA AND PHI.
1022 C      *****
1023      REAL U(NE),X(NE),Y(NE),KXI,KYI
1024      RAD=.17453293E-01
1025      TP=.6283185E+01
1026      TPS=TP*SIN(RAD*T)
1027      RP=RAD*P
1028      KXI=TPS*COS(RP)
1029      KYI=TPS*SIN(RP)

```

```
1030      SUM=0.0
1031      DO 1 K=1,NE
1032          PSI=KXI*I(K)+KYI*Y(K)
1033          SUM=SUM+U(K)*SIN(PSI)
1034 1      CONTINUE
1035      POW=SUM*SUM
1036      RETURN
1037      END
```

END

FILMED

DATE:

10-91

DTIC